CONVEX RELAXATIONS FOR CUBIC POLYNOMIAL PROBLEMS

A Thesis
Presented to
The Academic Faculty

by

Helder Inácio

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Industrial and Systems Engineering

Georgia Institute of Technology
May 2013

CONVEX RELAXATIONS FOR CUBIC POLYNOMIAL PROBLEMS

Approved by:

Shabbir Ahmed, Advisor
School of Industrial and Systems Engineering
Georgia Institute of Technology

Matthew J. Realff, Advisor
School of Chemical & Biomolecular Engineering
Georgia Institute of Technology

Jane C. Ammons
School of Industrial and Systems Engineering
Georgia Institute of Technology

Joel Sokol
School of Industrial and Systems Engineering
Georgia Institute of Technology

Kevin Norwood
Procter and Gamble

Date Approved: 04 February 2013

To the memory of my grandparents,

Beatriz da Conceição Ferreira,

Antonio Inácio,

Ana Mendes,

José Jesus Lourenço Rodrigues.

ACKNOWLEDGEMENTS

I would like to thank my advisers Dr. Shabbir Ahmed and Dr. Matthew Realff for the guidance, support, and encouragement during these long years. I also want to extend my gratitude to the members of my committee Dr. Jane Ammons, Dr. Kevin Norwood, and Dr. Joe Sokol for the valuable suggestions during my proposal and my defense. One of the points where Georgia Tech and particular ISyE excels is the quality of the faculty body. I would like to acknowledge those which I was lucky to learn from or to interact in some form. I also feel lucky to have been able to get help numerous times from Pam Morrison, Dr. Gary Parker, Trey Palmer and in the final stage of this process Dr. Paul Kvam. My undergraduate adviser Dr. Luís Nunes Vicente was my source of inspiration and encouragement to undertake this journey in the first place. Without his support I would not be here today. Stefan Vigerske was a great help when doing the computational experiments in SCIP. He helped me figure out many issues and challenges with the code implementation. Many new friends have come into my life in these years. From the first semester here I met friends that remained an essential part of my life during these years, either in school or out of it. Claudio Santiago, Daniel Faissol, Abhi Patel, and Eser Kirkizlar are in this group. Others appeared in my life later but have also been a source of many talks, good advice, and kind help, like Fatma Kilinç-Karzan, Bernardo Pagnoncelli and Jon "Pete" Petersen. I would like to also mention several other friends that are dear to my heart like Adaora Owko, Alejandro Toriello, Aly Megahead, Amandeep Parmar, Daniel Steffy, Feng Qiu, Huizhu "Crystal" Wang, Juan Pablo Vielma, Mike Hewitt, Norbert Remenyi, Ricardo Fukasawa, So Yeon Chun, Steve Tyber, Vijay Narayanan, Yao-Hsuan Chen and others that I shouldn't have forgotten. I was also lucky to meet people outside school whose friendship, help and support was greatly appreciated, among them Ling Liang, Sally and Tim Gago, and Keith Coleman. My family has loved me and supported me unconditionally in this journey and words are not enough to express by gratitude and love for my mother Maria

Rosa, my father David, my brother António and remaining family. Finally, I would like to acknowledge and thank the financial support received during these years in the form of either a teaching/research assistantship or the doctoral scholarship SFRH/BD/23016/2005 from the Portuguese funding institution FCT - Fundação para a Ciência e a Tecnologia.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

This dissertation addresses optimization of cubic polynomial problems. These are optimization problems for which at least one of the constraints (or the objective function) of the problem is a cubic polynomial and there exists no constraint with order higher than 3. Heuristics for finding good quality feasible solutions and for improving on existing feasible solutions for a complex industrial problem, involving cubic and pooling constraints among other complicating constraints, have been developed. The heuristics for finding feasible solutions are developed based on linear approximations to the original problem that enforce a subset of the original problem constraints while it tries to provide good approximations for the remaining constraints, obtaining in this way nearly feasible solutions. The performance of these heuristics has been tested by using industrial case studies that are of appropriate size, scale and structure. Furthermore, the quality of the solutions can be quantified by comparing the obtained feasible solutions against upper bounds on the value of the problem. Obtaining these upper bounds is an interesting problem by itself, and we have extended efficient existing techniques for bilinear problems for this class of cubic polynomial problems. Despite the efficiency of the upper bound techniques good upper bounds for the industrial case problem could not be computed efficiently within a reasonable time limit (one hour). We have applied the same techniques to subproblems with the same structure but about one fifth of the size and in this case, on average, the gap between the obtained solutions and the computed upper bounds is about 3%.

In the remaining part of the thesis we look at global optimization of cubic polynomial problems with non-negative bounded variables via branch and bound. A theoretical study on the properties of convex underestimators for non-linear terms which are quadratic in one of the variables and linear on the other variable is presented. A new underestimator is introduced for this class of terms. It is also shown that the straightforward polyhedral under approximation for terms of the form $-x^2y$ is indeed optimal, that is that that approximation

defines the convex hull of the term being approximated, although in the straightforward approximation has four constraints and we show that two of these suffice to define the convex hull. This means that there was redundancy in the straightforward approximation, which when eliminated reduces the size of problem possible allowing for a faster solving time.

The final part of the thesis describes the numerical testing of the previously mentioned underestimators together with approximations obtained by considering lifted approximations of the convex hull of the x^2y terms. Two sets of instances are generated for this test and the description of the procedures to generate the instances are detailed here. By analyzing the numerical results we can conclude that our proposed underestimator has the best behavior in the family of instances where the only non-linear terms present are of the form x^2y. This makes sense since this underestimator is developed specially for terms of this form. Problems originating from least squares are much harder to solve than the other class of problems. In this class of problems the efficiency of linear programming solvers plays a big role and on average the methods that use these solvers perform better than the others.

CHAPTER I

INTRODUCTION

In this thesis, we consider solving optimization problems involving cubic polynomials. These are polynomial optimization problems where at least one of the non-linear terms is cubic in the variables of the problem. Our study is motivated by the following industrial application. Chemicals (in this text we also use the terminology "raw materials") are first mixed to form a premix form of the chemicals that are in turn mixed in some pools. Each of the final products will extract its main composition from exactly one of these pools (the remaining fraction comes from the premix form of the chemicals directly). Such pooling problems have many applications in industry, notably in petroleum refining ([32, 20, 63, 71]) and wastewater treatment ([19]) among others. In the majority of the pooling problems in the literature there are linear requirements on the level of some chemicals in the pools and in the level of some chemicals in the final products. In our case besides these linear requirements the final products have other properties (or qualities) that must exceed certain thresholds or be close to target values. The variation of the properties with composition are in this case represented by cubic polynomial functions that arise from the design of experiments in which quadratic behavior in the original chemicals is not enough to capture the complex physics that maps the composition to the properties.

There is vast a literature that has proposed models and solution techniques for solving pooling problems. Some of the earliest references to the pooling problem are by Haverly with [48] and [49]. A survey of the recent advances in the pooling problem is given in ([64]). For references regarding the earlier work see ([38, 37, 14]). Our focus, when dealing with the aforementioned industrial problem, is to obtain good quality solutions in a reasonable computational time. For this reason, when dealing with the pooling constraints, we use a technique similar to the one introduced by Pham, Laird, and El-Halwagi in ([69]) which consists in picking one of the variables in each bilinear term and forcing each of these

variables to assume one of finitely many values. To be more precise, given the interval of possible values for the chosen variable in a bilinear term, we pick a finite number of points in the interval and force the variable to take one of these values using binary variables. We consider two main versions for the problem to be solved: in one version we try to minimize overall the cost while imposing lower bounds on the levels of the qualities monitored. In the other version we keep the overall cost of the product portfolio at a specified level and maximize a weighted average of the product qualities considered. In any case all of these problems are categorized as a *Mixed Integer Non-Linear Problem* (**MINLP**).

We focus on the second version of the problem and develop heuristics to find good quality solutions quickly. This is done by building *Mixed Integer Linear Problem* (**MILP**) approximations to the problem which provide nearly feasible solutions to the non-linear problem. These solutions can then be used as the starting point of specialized solvers which, without these starting points, would not even be able to provide a feasible solution to the problem. The feasible solutions to the **MINLP** problem obtained in this fashion are not, in general, global optimal solutions. So we develop a produce to improve this procedure by means of a local search where only a subset of the raw materials is considered at a time.

It is also important to be able to evaluate the quality of the solutions obtained. Given a cubic optimization function with bounded variables, it is well known that we can obtain lower and upper bounds for this cubic function using techniques introduced by McCormick in [62]. However, the bounds obtained using this technique are usually far from the true lower and upper bounds, so further refinement is necessary to obtain better bounds. In our case we do this by picking some of the variables in the problem and partitioning the original range of these variables in smaller intervals. By using binary variables we enforce that the original variable must be in exactly one of the smaller intervals, which gives tighter bounds on the original expression. This extends the work done in ([86, 45]) which considers bilinear optimization problems.

The problem aforementioned has several interesting and challenging features among them the pooling constraints, the combinatorial structure and the polynomial cubic functions that in the general setting are to be imposed as lower bounds in the final products.

In the next part of the thesis we focus on global optimization of cubic polynomial problems with non-negative bounded variables via branch and bound methods. In order to perform the branch and bound procedure it is necessary to compute a lower bound for the problem corresponding to each node. The lower bounds that we consider are obtained by relaxing each of the non-linear terms in the constraints and objective function and adding the relaxation corresponding to each of the non-linear terms to the problem. The possible non-linear terms for a cubic polynomial problem in variables x, y, z are bilinear terms (xy), trilinear terms (xyz), quadratic terms (x^2), cubic terms (x^3), and terms which are quadratic in one variable and linear on the other variable (x^2y). Multilinear terms are well studied in the literature and good approximations for quadratic and cubic terms are also well known since these terms are convex for non-negative variables. So we focus on studying approximations for terms which are linear in one of the variables and quadratic in the other. The convex hull for these types of terms has been previously been described in ([80]) but the obtained explicit function of variables x and y is complex and cannot be used directly in optimization software. The convex hull admits, however, a conic quadratic or semidefinite representation, so relaxations for these problems can be obtained by using specialized solvers for second order cubic problems or semidefinite programming solvers. We first study the theoretical properties of different underestimators for these terms, among them a new class of underestimators. These new proposed underestimators are defined by convex quadratic functions so they can be used in general-purpose non-linear solvers when solving convex relaxations for the original cubic problem. Finally, we perform a computational study where we compare how the different approximations behave in a branch and bound setting. Since, as mentioned before, the convex hull cannot be easily expressed in terms of the original variables, but admits a conic quadratic representation, we use the lifted approximation of the second order cone developed by Nemirovski and Ben-Tal in ([24]) to build a linear approximation for the convex hull. Additional constraints and variables are necessary to define this approximation, but the number of additional constraints and variables is at most polynomial in the inverse of the desired accuracy. The three methods under comparison are the proposed convex quadratic underestimators, the linear approximation that would be obtained when

reformulated via introduction of new variables, and the underestimators obtained by considering the lifted approximation of the convex hull. These different methods are compared across two randomly generated families of instances. The problems in one of the families have only linear terms and non-linear terms of the form x^2y while the instances in the other family of problems are generated by means of the least square estimation procedure.

The remainder of this thesis is organized as follows. In Chapter 2 we describe the features of the aforementioned industrial problem, the heuristics developed for obtaining solutions for the problem, and numerical results for the proposed methods. In Chapter 3 we develop upper bounds for the problem and give numerical results concerning these upper bounds. In Chapter 4 we present a new underestimator for terms of the form x^2y and study the numerical properties of this class of underestimators, comparing it against other underestimators for the same type of terms. Chapter 5 gives the details and properties of the lifted linear approximations and it describes details of the implementation of the different approximations. We also describe the generation of the two families of instances, and we present and discuss the computational results. In Chapter 6 we summarize the contributions of this research and future directions.

CHAPTER II

HEURISTICS FOR A POOLING PROBLEM WITH CUBIC PROPERTIES

2.1 Introduction

The manufacturing of complex chemical products requires precise blending of a large number of ingredients that impart different properties to the product with relatively different efficacy. For each final product, there is a set of properties that we are interested in either maximizing or keeping at an acceptable level. In general these properties are given by polynomial cubic functions in the raw material composition, which arise from standard cubic polynomial fitting of a set of test points/responses placed in the composition space by a design of experiments methodology. The differences in the unit price of the ingredients creates an optimization opportunity to achieve the lowest possible cost for a given property value, or to have the highest property value for a given cost. This picture is further complicated by the blending of certain raw materials in large pools prior to their inclusion in the product. In the problem examined in the thesis, each final product uses exactly one of the pools for the majority of the volume in its composition and it is differentiated from the other products sharing the same pool by the fraction of the composition that is extracted from the pool and by the direct addition of raw materials. Figure 1 gives a visual representation of this process.

2.2 Short Literature Review

There are two key classes of constraints that make our formulation of the blending problem complex. The first are the "pooling constraints" that appear in any problem where the property of the material exiting a blending tank must be computed by a weighted average of the incoming material flows and properties. These have been extensively studied: see the book [82] for details and references of early work and [64] for a review of advances made since 2000. In [69] the authors introduce the notion of discretizing the variable appearing

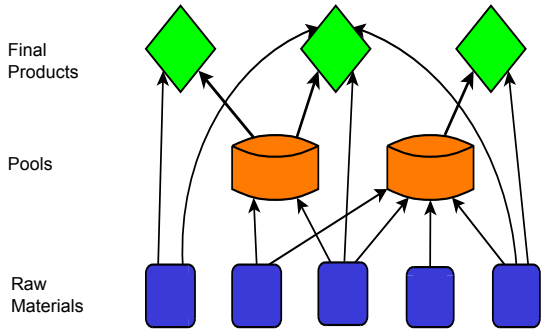

Figure 1: Diagram of Chemical Flow

in the nonconvex bilinear pooling equalities. We integrate this approach in the framework of our problem. In [86] the authors discuss several piecewise linear approximations for bilinear problems. In [45] the authors present a computational study of this piecewise linear approximations for bilinear problems. To be able to handle the second class of constraints, the cubic property constraints, we extend some of the piecewise approximations in these works.

2.3 Problem Notation

In Table 1 we introduce the notation used in the description of the model. Some notation pertaining only to specific parts of the problem will be introduced where appropriate.

We use the notation $P_{jk}(x_{\cdot j})$ to denote the value of property k in product j, with $k \in QA$ and $j \in FP$. $x_{\cdot j} = (x_{1j}, x_{2j}, \ldots, x_{N_{RM}j})$ denotes the vector in which the first component varies in the allowed set but the second component is fixed. This notation is similar to **Matlab** notation. In the most general form these property functions are cubic in $x_{\cdot j}$.

$$P_{jk}(x_{\cdot j}) = a^k + \sum_i b_i^k x_{ij} + \sum_{i_1,i_2} c_{i_1 i_2}^k x_{i_1 j} x_{i_2 j} + \sum_{i_1,i_2,i_3} d_{i_1 i_2 i_3}^k x_{i_1 j} x_{i_2 j} x_{i_3 j} \tag{1}$$

and arise from the use of a response surface methodology to empirically fit the experimental results of benchscale blending. In appendix A we provide some of the coefficients used in the problem, or some approximations for the coefficients used in the problem.

In this work we consider two mathematical formulations: **AllConstrained** and **MaxAvg**.

Table 1: Nomenclature for the Problem

Indices	
$i \in \{1, \ldots, N_{RM}\} \equiv I^{RM}$	Raw materials
$l \in \{1, \ldots, N_P\} \equiv I^P$	Pools
$j \in \{1, \ldots, N_{FP}\} \equiv FP$	End products
$k \in \{1, \ldots, N_A\} \equiv QA$	Attributes (qualities monitored)
$s \in \{1, \ldots, S\}$	Points considered to discretize pooling
Sets	
T_Z	pairs for which the connection (i, j) exists
T_C	indexes in the cleaning model
T_S	indexes in the stability model
Variables	
x_{ij}	Fraction of i in output j
y_{lj}	Fraction from pool l in output j
z_{ij}	Fraction of i going directly to j
q_{il}	Proportion of flow i in pool l
$P_{jk}(\cdot)$	Level of quality attribute k in product j
w_{lj}	Binary indicating if j extracts from l or not
v_{ilj}	Fraction of i in j through l
Parameters	
c_i	Cost of raw material i
D_j	Demand of end product j
T_i^L, T_i^U	Bounds for the usage of raw material i
$\bar{P}_{jk}$	Desired level of quality k in product j
ω_{jk}	Weights associated with end product j and quality k
y_{lj}^s	Value for the discretized point s for brand j and pool l

A mathematical formulation for **AllConstrained** is given in Equation 2.

$$
\begin{aligned}
\text{Minimize} \quad & \sum_{j \in FP} D_j \sum_{i \in I^{RM}} c_i x_{ij} \\
\text{Subject to :} \quad & \sum_{i \in I^{RM}} x_{ij} = 1 && \forall j \in FP \\
& \sum_{i \in I^{RM}} q_{il} = 1 && \forall l \in I^P \\
& \sum_{l \in I^P} w_{lj} = 1 && \forall j \in FP \\
& \sum_{l \in I^P} v_{ilj} + z_{ij} = x_{ij} && \forall i \in I^I, j \in FP \\
& y_{lj} \leq w_{lj} && \forall l \in I^P, j \in FP \\
& T_i^L \leq \sum_{j \in FP} D_j x_{ij} \leq T_i^U && \forall i \in I^{RM} \\
& \sum_{i \in I^{RM}} v_{ilj} = y_{lj} && \forall l \in I^P, j \in FP \\
& P_{jk}(x_{\cdot j}) \geq \bar{P}_{jk} && \forall j \in FP, k \in T_C \cup T_S \\
& q_{il} \cdot y_{lj} = v_{ilj} && \forall i \in I^{RM}, l \in I^P, j \in FP \\
& x \geq 0,\ q \geq 0,\ v \geq 0,\ w \text{ binary}
\end{aligned}
\tag{2}
$$

A mathematical formulation for **MaxAvg** is given in Equation 3.

$$
\begin{aligned}
\text{Maximize} \quad & \sum_{j \in FP, k \in T_C} \omega_{jk} P_{jk}(x_{\cdot j}) \\
\text{Subject to :} \quad & \sum_{i \in I^{RM}} x_{ij} = 1 \qquad \forall j \in FP \\
& \sum_{i \in I^{RM}} q_{il} = 1 \qquad \forall l \in I^P \\
& \sum_{l \in I^P} w_{lj} = 1 \qquad \forall j \in FP \\
& \sum_{l \in I^P} v_{ilj} + z_{ij} = x_{ij} \qquad \forall i \in I^I, j \in FP \\
& y_{lj} \leq w_{lj} \qquad \forall l \in I^P, j \in FP \\
& T_i^L \leq \sum_{j \in FP} D_j x_{ij} \leq T_i^U \qquad \forall i \in I^{RM} \\
& \sum_{i \in I^{RM}} v_{ilj} = y_{lj} \qquad \forall l \in I^P, j \in FP \\
& P_{jk}(x_{\cdot j}) \geq \bar{P}_{jk} \qquad \forall j \in FP, k \in T_S \\
& q_{il} \cdot y_{lj} = v_{ilj} \qquad \forall i \in I^{RM}, l \in I^P, j \in FP \\
& \sum_{j \in FP} D_j \sum_{i \in I^{RM}} c_i \cdot x_{ij} \leq \bar{c} \\
& \bar{c}_j^L \leq D_j \sum_{i \in I^{RM}} c_i \cdot x_{ij} \leq \bar{c}_i^U \qquad \forall j \in FP \\
& x \geq 0,\ q \geq 0,\ v \geq 0,\ w \text{ binary}
\end{aligned}
\tag{3}
$$

Together with problem described in Equation 2 we consider another problem, described in Equation 3.

We now describe in detail the meaning of each group of constraints in the two problems **AllConstrained** and **MaxAvg**.

$\sum_{j \in FP} D_j \sum_{i \in I^{RM}} c_i x_{ij}$

$\sum_{i \in I^{RM}} c_i x_{ij}$ represents the cost of producing one unit of final product j, so the expression represents to total cost of producing the portfolio for the considered production term.

$\sum_{i \in I^{RM}} x_{ij} = 1 \quad \forall j \in FP$

The sum of the composition of each final product must add to 1, in relative terms.

$\sum_{i \in I^{RM}} q_{il} = 1 \quad \forall l \in I^P$

The composition of each pool must add to 1, in relative terms.

$$\sum_{l \in I^P} w_{lj} = 1 \quad \forall j \in FP$$

Each final product must be associated to exactly of the existing pools.

$$y_{lj} \leq w_{lj} \quad \forall l \in I^P, j \in FP$$

If a final product is not associated to a pool ($w_{lj} = 0$) then no fraction of that final product can come from that pool.

$$T_i^L \leq \sum_{j \in FP} D_j x_{ij} \leq T_i^U \quad \forall i \in I^{RM}$$

These represent upper and lower limits in the amount of raw materials that can be used during the production term. This constraint may arise from the fact that specific raw materials may be hard to manufacture, or subject to seasonal restrictions. If the usage of a raw material is not subject to the lower or upper bound usage restriction we simply have $T_i^L = 0$ or $T_i^U = \infty$.

$$y_{lj} = \sum_{i \in I^{RM}} v_{ilj} \quad \forall l \in I^P, j \in FP$$

For each pool and each final product the total fraction coming from the pool to the final product, y_{lj}, is the sum of the individual fractions of the raw materials in the pool going to the final product (v_{ilj}).

$$P_{jk}(x_{.j}) \geq \bar{P}_{jk} \quad \forall j \in FP, k \in T_C \cup T_S$$

Nonlinear inequalities representing the property constraints to be met by the final product.

$$v_{ilj} = q_{il} \cdot y_{lj} \quad \forall i \in I^{RM}, l \in I^P, j \in FP$$

This set of equalities is known as the pooling constraints, and it means that the fraction of raw materials coming from a specific pool must be at the same proportion to the proportion of those raw materials in the pool.

$$\sum_{j \in FP} D_j \sum_{i \in I^{RM}} c_i \cdot x_{ij} \leq \bar{c}$$

This constraint represents an upper bound on the total cost of the portfolio solution.

$$\bar{c}_j^L \leq D_j \sum_{i \in I^{RM}} c_i \cdot x_{ij} \leq \bar{c}_i^U \quad \forall j \in FP$$

This constraint imposes an upper bound on the cost on individual final products.

Problems in Equations 2 and 3 are closely related; in problem **MaxAvg** we maximize a cubic function obtained by averaging the cubic functions defining the right hand side of the property constraints using specific weights for each of the property constraints. Only the property functions in the set T_C contribute for the definition of the objective function, the property constraints defined over the set T_S are left explicitly in **MaxAvg** model. The objective function of model **AllConstrained** is now kept under a reasonable value by adding explicit constraints imposing lower and upper bounds. Additionally we also impose bounds on the individual cost of each final product.

2.3.1 Features of Problem not Explicit in Formulations

A feature of the problem that we have not mentioned explicitly in the description above is that although we describe the property constraints as explicit functions of the x variables, the implementation involves an extra group of variables. Recall that the property constraints are divided in two groups T_C and T_S. Constraints in the set T_S are always explicitly enforced in the models we present, while constraints in the set T_C may be moved into the objective function and small violations may be accepted in the solutions produced. The non-linear functions in the set T_S are explicit polynomials of the variables x and are in the most general form quadratic polynomials on x. For the functions in the set T_C the situation is different. In this case the polynomials are actually explicit function of variables ξ where the variables ξ are defined as an affine transformation of the variables x:

$$\xi_{ik} = \frac{x_{ik} - \gamma_{ik}}{\mu_{ik}} \tag{4}$$

In Equation 4 γ_{ik} and μ_{ik} are constants defining the affine transformation. The definition of these extra variables allows us to simplify the process of writing the constraints associated with set T_C. The number of variables in the polynomials of both sets, that is the variables for which there exists at least one polynomial in each of the sets with a non-zero coefficient, is very small. There are also additional variables present in the definition of the property

constraints in set T_C that have no relation with the chemical composition. These correspond to extra factors like temperature which we keep constant in the models considered but which would be harder to incorporate without the definition of these additional variables.

2.4 Primal Heuristics

2.4.1 Finding Feasible Solutions

In this section we describe heuristics for problem described in Equation 3 (**MaxAvg**). Note that if the pooling constraints and the property constraints were not present the problem described in Equation 3 would be a mixed integer linear problem (**MILP**). We build an associated problem in which we replace these constraints by linear approximations. This linear approximation provides a solution that is feasible with respect to all the linear constraints in the original problem, but depending on how we build the linear approximations might not be feasible for the original pooling and property constraints.

2.4.2 Pooling Constraints

2.4.2.1 Earlier Work

Pham et al in [69] use discretization of the pooling qualities to obtain good quality solutions for the pooling problem. The constraints considered are of the form

$$b_{jq} \cdot \sum_{i=1}^{l} X_{ij} = \sum_{i=1}^{l} X_{ij} \cdot a_{iq} \quad j = 1, \ldots, m,\ q = 1, \ldots N_q \tag{5}$$

In Equation 5 l is the number of inputs, m is the number of pools and N_q is the number of qualities in the problem; a_{iq} is a parameter describing the level of quality q at input i, X_{iq} and b_{jq} are variables representing, respectively, the flow rate (or amount) from source i to pool j and the level of quality q at pool j. The discretization is done by considering only discrete values for the level of each of the qualities q at pool j, b_{jq}. For each quality q suppose we consider t_q intervals in the discretization of q, which gives a total of t_q+1 points for each of the qualities q, $q = 1, \ldots N_1$. For each quality q the range of admissible values, $[b_q^{min}, b_q^{max}]$ is replaced by a finite set of values $\{b_q^1 = b_q^{min}, b_q^2, \ldots, b_q^{t_q+1} = b_q^{max}\}$. Enumerating all possible values for the qualities as $b_1^1, b_1^2, \ldots, b_1^{t_1+1}, b_2^1, \ldots, b_{N_q}^{t_{N_q}+1} = b_1, b_2 \ldots b_M$, with M given by $M = (t_1+1) \cdot (t_2+1) \cdot \cdots \cdot (t_{N_q+1})$ equalities in Equation 5 are replaced with the constraints

described in Equation 6:

$$\begin{aligned} b_{jq} \cdot \sum_{i=1}^{l} X_{ij} &= \sum_{i=1}^{l} X_{ij} \cdot a_{iq} && j = 1, \ldots, M \\ f_j &\in \{0, 1\} && j = 1, \ldots, M \\ L_j \cdot f_j \leq \sum_{i=1}^{l} X_{ij} &\leq U_j \cdot f_j && j = 1, \ldots, M \\ \sum_{j=1}^{M} X_{ij} &\leq m \end{aligned} \tag{6}$$

Pham et al also consider a variant of the problem where instead of discretizing the possible values for the amount of quality q in pool j, b_{qj}, these possible values are discretized implicitly by considering the attainable values resulting from the quality levels at the sources. For this new variables x_{ij} representing the flow rate proportion from source i to pool j are defined by

$$x_{ij} = \frac{X_{ij}}{\sum_{i=1}^{l} X_{ij}} \tag{7}$$

Then b_{jq} is defined by

$$\begin{aligned} b_{jq} &= \sum_{i=1}^{l} x_{ij} \cdot a_{iq} \\ 1 &= \sum_{i=1}^{l} x_{ij} \end{aligned} \tag{8}$$

In Equation 8, $\sum_{i=1}^{l} x_{ij} = 1$ is a convex hull condition, and in this model the variables x_{ij} are now discretized. This model allows to reduce the range of values considered for each b_{jq}.

2.4.2.2 Our Approach

The pooling constraints in problem from Equation 3 are defined by

$$v_{ilj} = q_{il} \cdot y_{lj} \quad \forall i \in I^{RM}, l \in I^{P}, j \in FP \tag{9}$$

To simplify the exposition we consider now a generic version of these constraints

$$z = x \cdot y \tag{10}$$

with all variables nonnegative and bounded, that is $0 \leq x^l \leq x \leq x^u \leq 1$ and $0 \leq y^l \leq y \leq y^u \leq 1$. When necessary we will again revert to the notation of our original problem. The

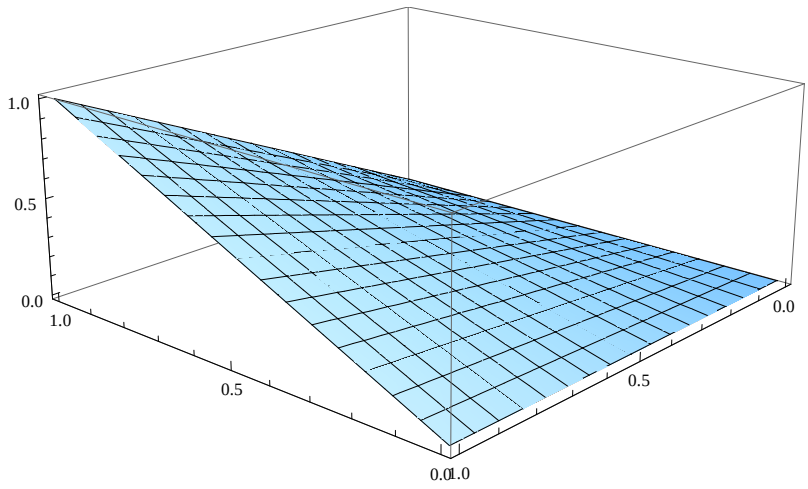

Figure 2: Product of x and y in the unit box

pooling constraints are a main source of non-convexity in the problem. A figure representing the surface obtained by plotting $z = x \cdot y$ with x in y in $[0, 1]^2$ is represented in Figure 2.

Restrictions for this type of constraints are well studied in the literature. Some of the earliest references to the pooling problem are by Haverly in [48] and [49]. In these formulations tiny instances were studied by means of a recursion technique, which consists of estimating some parameters in the problem, and assuming these parameters fixed, solving the resulting model, a linear program, and depending on the solution either adjusting again the parameters and solving the model or stopping the process. The techniques used in the recursion model fall generally under an approach denominated by **SLP** (*Sequential Linear Programming*) in which a linearized version of the pooling problem is obtained by assuming some of the parameters to be constant, this version of the problem is then solved and the model updated. In [46] Harvey J. Greenberg gives an overview of this process and also sensitivity analysis and infeasibility diagnosis for **SLP**. To obtain global solutions to the pooling problem, we can use a spatial branch and bound method (see Horst and Tuy [52]) which uses the lower and upper bounds on the variables to generate piecewise linear concave and convex approximations to each bilinear term. Convex hull relaxations for bilinear terms were first introduced by McCormick in [62] and the proof that these functions define the convex hull for bilinear terms is by Al-Khayyal and Falk in [17]. Al-Khayyal and Sherali

discuss finite termination of these types of algorithms in [18]. The drawback with these approaches is that in general they only provide feasible solutions in the limit, that is at the $n-th$ iteration of the procedure we can have lower and upper bounds l^n and u^n such that $l^n \leq x \cdot y \leq u^n$ and $\lim_n l^n = \lim_n u^n$ but $l^n < u^n$ for any n. We use a technique similar to the one used in [69] in which one of the variables is restricted to take values from a finite set. Say we pick variable y and allow it to only take values from a finite subset of values, $S_y = \{\bar{y}^1, \bar{y}^2, \ldots, \bar{y}^N\}$. We call variable y the discretized variable. With each of the points of S_y, $\bar{y}^i$ we associate a binary variable y^i and add the following constraints to the model

$$\begin{aligned} x \cdot \bar{y}^i - (1 - y^i) &\leq z, \quad i = 1, \ldots, N \\ x \cdot \bar{y}^i + (1 - y^i) &\geq z, \quad i = 1, \ldots, N \\ \textstyle\sum_{i=1}^N y^i \cdot \bar{y}^i &= y \\ \textstyle\sum_{i=1}^N y^i &= 1 \end{aligned} \tag{11}$$

To see how the constraints in Equation 11 actually imply that the bilinear equations are satisfied note that if $y^i = 0$ then the two first inequalities in Equation 11 are always satisfied since $0 \leq x, y \leq 1$, so we have $0 \leq x \cdot \bar{y}^i \leq 1$ for any $i = 1, \ldots, N$ which implies

$$x \cdot \bar{y}^i - (1 - y^i) = x \cdot \bar{y}^i - 1 <= 1 - 1 = 0 <= z \tag{12}$$

so it shows that the first inequality becomes redundant for i such that $y^i = 0$. Same reasoning applies for the second inequality in Equation 11. When $y^i = 1$ it is clear that $(1-y^i) = 0$ so the two first inequalities of Equation 11 became $z \geq x \cdot \bar{y}^i$ and $z \leq x \cdot \bar{y}^i$ which is equivalent to $z = x \cdot \bar{y}^i$. This means that the constraints in Equation 11 actually define a big-M type formulation with $M = 1$. For each equality described in Equation 10 we add N additional variables and $2N + 2$ constraints. For each equality of the type described in Equation 10 we add $2N$ constraints to the model and for each variable that we choose to partition we add N variables and 2 extra constraints to the model. We have therefore to be careful with the choice of the variable to discretize. In our case the family of equalities in which we use this technique is

$$v_{ilj} = q_{il} \cdot y_{lj} \quad \forall i \in I^{RM}, l \in I^P, j \in FP \tag{13}$$

Assume now that we choose variables y_{lj} to be the ones to be partitioned. Remember that these variables represent the fraction of pool l going into final product j, and while in principle the lower and upper are 0 and 1, in practice the actual bounds of the variables may not be the complete $[0, 1]$ range, so we choose a range for the variables that is still big enough to explore new possibilities but that allows to reduce the original range of the variables so using the same number of points for the discretization gives a better approximation since the distance between the points in the approximation is smaller. We have to also take into account that in our problem each final product must extract from exactly one pool, so we must have $y_{lj} = 0$ if final product j is not associated with pool l. This is represented by the constraints in Equation 14.

$$\begin{aligned} \sum_{l \in I^P} w_{lj} &= 1 \qquad \forall j \in FP \\ y_{lj} &\leq w_{lj} \quad \forall l \in I^P, j \in FP \end{aligned} \tag{14}$$

Since we are assuming that the variables y_{lj} have positive lower bounds $0 < y_{lj}^L \leq y_{lj} \leq y_{lj}^U \leq 1$, this actually means that the variables y_{lj} are semi-continuous variables in our model, that is

$$y_{lj} \in \{0\} \cup [y_{lj}^L, y_{lj}^U], \quad l \in I^P, j \in FP \tag{15}$$

One possible technique to handle this is to always add 0 to set of points of the variable to be discretized so for each y_{lj} variable we would have

$$\left\{\bar{y}_{lj}^1, \bar{y}_{lj}^2, \ldots, \bar{y}_{lj}^{N_{lj}}\right\} = \left\{0, y_{lj}^L, \ldots, y_{lj}^U\right\} \tag{16}$$

Defining $S_{lj} = \{1, \ldots, N_{jl}\}$ we add the following constraints to the model

$$\begin{aligned} q_{il} \cdot \bar{y}_{lj}^s - (1 - y_{lj}^s) &\leq v_{ilj}, \quad s \in S_{lj}, l \in I^P, j \in FP \\ q_{il} \cdot \bar{y}_{lj}^s + (1 - y_{lj}^s) &\geq v_{ilj}, \quad s \in S_{lj}, l \in I^P, j \in FP \\ \textstyle\sum_{s \in S_{lj}} y_{lj}^s &= 1, \qquad \forall l \in I^P, j \in FP \\ \sum_{l \in I^P} (1 - y_{lj}^1) &= 1, \qquad j \in FP \end{aligned} \tag{17}$$

We now show that this can be improved by adding instead the following set of constraints

to the model:

$$\begin{aligned} q_{il} \cdot \bar{y}^s_{lj} - (1 - y^s_{lj}) &\leq v_{ilj}, \quad s \in S_{lj}, l \in I^P, j \in FP \\ q_{il} \cdot \bar{y}^s_{lj} + (1 - y^s_{lj}) &\geq v_{ilj}, \quad s \in S_{lj}, l \in I^P, j \in FP \\ \sum_{s \in S_{lj}} y^s_{lj} &= w_{lj}, \quad l \in P, j \in FP \end{aligned} \tag{18}$$

In this case we do not need to include 0 in the set of points considered in the discretized variable, so for each pair (l, j) we have $\{\bar{y}^1_{lj}, \bar{y}^2_{lj}, \ldots, \bar{y}^{N_{lj}}_{lj}\} = \{y^L_{lj}, \ldots, y^U_{lj}\}$. If $w_{lj} = 0$ then $\sum_{s \in S_{lj}} y^s_{lj} = w_{lj} = 0$ which implies that each for each of the y^s_{lj} variables are 0, while if $w_{lj} = 1$ exactly one of the variables y^s_{lj} is 1 which forces the corresponding bilinear equality to be satisfied with equality.

We have also that if $w_{lj} = 0$ then $y_{lj} \leq w_{lj}$ and $\sum_{i \in I^{RM}} v_{ilj} = y_{lj}$ imply $v_{ilj} = 0$, so all the bilinear constraints are in fact satisfied by the model. In this work we consider the points to be equally spaced, (the exception to this may be if when we consider 0 to be one of the admissible points as discussed above).

2.4.3 Property Constraints

Property constraints in model **AllConstrained** are given by

$$P_{jk}(x_{\cdot j}) \geq \bar{P}_{jk} \quad \forall j \in FP, k \in T_C \cup T_S \tag{19}$$

while in model **MaxAvg** in Equation 3 we maintain only the constraints in the set T_S explicitly in the model while the constraints

$$P_{jk}(x_{\cdot j}) \geq \bar{P}_{jk} \quad \forall j \in FP, k \in T_C \tag{20}$$

are now part of the objective function where a weight ω_{jk} is associated with each of the pools j and final product k, giving the following objective function to be maximized:

$$\sum_{j \in FP, k \in T_C} \omega_{jk} P_{jk}(x_{\cdot j}) \tag{21}$$

The weights in Equation 21 allow to differentiate the different property constraints which allows to target products where specific property are specially high or correct a solution not meeting the standards for some property by increasing the weight associated with that property.

We note now that a model simply trying to minimize the deviation from the required targets each of some of the property constraints would not be advantageous from a computational point of view, the main candidates for the objective function of this model would be

$$\sum_{j \in FP, k \in T_C} \left| P_{jk}(x_{\cdot j}) - \bar{P}_{jk}(x_{\cdot j}) \right| \tag{22}$$

and

$$\sum_{j \in FP, k \in T_C} \left(P_{jk}(x_{\cdot j}) - \bar{P}_{jk}(x_{\cdot j}) \right)^2 \tag{23}$$

The objective function in Equation 22 introduces discontinuities and not all solvers in **GAMS** handle this type of function efficiently while the objective function in Equation 23 would have many high order terms (it would be a polynomial of degree six) which is not handled efficiently by current state of the art non-linear solvers.

In order to build a linear approximation to **MaxAvg** model we build linear approximations to the property functions; in this case given a nonlinear function P_{jk} we consider the first order approximation $\tilde{P}_{jk}(x_{\cdot j}; \tilde{x})$ around a given point $\tilde{x}$

$$\begin{aligned} \tilde{P}_{jk}(x_{\cdot j}; \tilde{x}) &= P_{jk}(\tilde{x}) + \sum_{i=1}^{n} \frac{\partial P_{jk}(\tilde{x})}{\partial x_{ij}} (x_{ij} - \tilde{x}_{ij}) - \varepsilon_{jk} \\ &= P_{jk}(\tilde{x}) + \nabla P_{jk}(\tilde{x})' (x_{\cdot j} - \tilde{x}_{\cdot j}) - \varepsilon_{jk} \end{aligned} \tag{24}$$

In equation 24 ε_{jk} is a constant term that is added to first order approximation of the function P_{jk}, and it is useful if we want to try to make a specific property function feasible, so we have $\varepsilon_{jk} \geq 0$ for $k \in T_S$, while in the case of the property functions that we move to the objective function it does not make sense to add any constant term so in this case $\varepsilon_{jk} = 0$ for $k \in T_C$. Recalling from Equation 1 that in general

$$P_{jk}(x_{\cdot j}) = a^k + \sum_i b_i^k x_{ij} + \sum_{i_1, i_2} c_{i_1 i_2}^k x_{i_1 j} x_{i_2 j} + \sum_{i_1, i_2, i_3} d_{i_1 i_2 i_3}^k x_{i_1 j} x_{i_2 j} x_{i_3 j} \tag{25}$$

we have that

$$\begin{aligned} \frac{\partial P_{jk}(x)}{\partial x_{ij}} = \; & b_i^k + \sum_{r \neq i} c_{ri}^k x_{rj} + 2 c_{ii}^k x_{ij} + \sum_{r \neq i, s \neq i} d_{irs}^k x_{rj} x_{sj} + \\ & 2 x_{ij} \sum_{r \neq i} d_{iir}^k x_{rj} + 3 d_{iii}^k x_{ij}^2 \end{aligned} \tag{26}$$

2.4.4 Correcting Benchmark Solution

Often it is possible to obtain a "good" initial solution to the problem through heuristics. Call this solution $\bar{x}^0$. This solution might not satisfy all the constraints of the problem, but we are still interested in comparing our solution against it. In particular we have

$$\bar{P}_{jk} \approx P_{jk}(\bar{x}^0) \tag{27}$$

The main issue with the heuristic solution $\bar{x}^0$ is that it does not satisfy all the constraints in models **MaxAvg** and **AllConstrained**, so we cannot use it as an initial solution for these models. Additionally we want to use a benchmark solution to compare against the solutions we obtain, and if the benchmark solution is not feasible to the model the comparison is not fair. We correct this by finding a solution that is close to the provided solution. In order to do this we construct auxiliary models **FindCloseLin** and **FindCloseNonLin** defined in Equations 28 and 29.

$$\begin{aligned}
\text{Minimize} \quad & \left\|\bar{x}^0 - x\right\|_1 \\
\text{Subject to :} \quad & \sum_{i \in I^{RM}} x_{ij} = 1 && \forall j \in FP \\
& \sum_{i \in I^{RM}} q_{il} = 1 && \forall l \in I^P \\
& \sum_{l \in I^P} w_{lj} = 1 && \forall j \in FP \\
& \sum_{l \in I^P} v_{ilj} + z_{ij} = x_{ij} && \forall i \in I^I, j \in FP \\
& y_{lj} \leq w_{lj} && \forall l \in I^P, j \in FP \\
& T_i^L \leq \sum_{j \in FP} D_j x_{ij} \leq T_i^U && \forall i \in I^{RM} \\
& \sum_{i \in I^{RM}} v_{ilj} = y_{lj} && \forall l \in I^P, j \in FP \\
& q_{il} \cdot \bar{y}_{lj}^s - (1 - y_{lj}^s) \leq v_{ilj}, && s \in S_{lj}, l \in I^P, j \in FP \\
& q_{il} \cdot \bar{y}_{lj}^s + (1 - y_{lj}^s) \geq v_{ilj}, && s \in S_{lj}, l \in I^P, j \in FP \\
& \sum_{s \in S_{lj}} y_{lj}^s = w_{lj}, && l \in P, j \in FP \\
& x \geq 0,\ q \geq 0,\ v \geq 0,\ w \text{ binary}
\end{aligned} \tag{28}$$

$$
\begin{aligned}
&\text{Minimize} && \left\|\bar{x}^0 - x\right\|_1 && \\
&\text{Subject to :} && \sum_{i \in I^{RM}} x_{ij} &= 1 && \forall j \in FP \\
& && \sum_{i \in I^{RM}} q_{il} &= 1 && \forall l \in I^P \\
& && \sum_{l \in I^P} w_{lj} &= 1 && \forall j \in FP \\
& && \sum_{l \in I^P} v_{ilj} + z_{ij} &= x_{ij} && \forall i \in I^I, j \in FP \\
& && y_{lj} &\leq w_{lj} && \forall l \in I^P, j \in FP \\
& && T_i^L \leq \sum_{j \in FP} D_j x_{ij} &\leq T_i^U && \forall i \in I^{RM} \\
& && \sum_{i \in I^{RM}} v_{ilj} &= y_{lj} && \forall l \in I^P, j \in FP \\
& && q_{il} \cdot y_{lj} &= v_{ilj} && \forall i \in I^{RM}, l \in I^P, j \in FP \\
& && \sum_{s \in S_{lj}} y_{lj}^s &= w_{lj}, && l \in P, j \in FP \\
& x \geq 0,\ q \geq 0,\ v \geq 0,\ w \text{ binary} &&
\end{aligned}
\tag{29}
$$

Regarding the objective function in the models from Equations 28 and 29 we note that it is actually implemented by introducing auxiliary variables t_{ij} for $i \in I^{RM}$ and $j \in FP$ and by adding the following constraints to the model:

$$
\begin{aligned}
t_{ij} &\geq \bar{x}_{ij}^0 - x_{ij} && i \in I^{RM}, j \in FP \\
t_{ij} &\geq -(\bar{x}_{ij}^0 - x_{ij}) && i \in I^{RM}, j \in FP
\end{aligned}
\tag{30}
$$

and with these constraints the objective function to minimize, $\left\|\bar{x}^0 - x\right\|_1$, is actually written as $\sum_{i \in I^{RM}, j \in FP} t_{ij}$.

The procedure to correct the initial solution is described in detail in Algorithm 1.

input : Initial Solution: $\bar{x}^0, \bar{v}, \bar{q}$

output: Corrected Solution: $\bar{x}, \bar{v}, \bar{q}$

$x^c \leftarrow \bar{x}^0, \quad v^c \leftarrow \bar{v}, \quad q^c \leftarrow \bar{q}, \quad y^L \leftarrow \bar{y}^L \quad y^U \leftarrow \bar{y}^U$

$\bar{y}^s_{lj} \leftarrow y^L_{lj} + (s-1)(y^U_{lj} - y^L_{lj})/(|S_{lj}| - 1)$

for $l \in I^P$ **do**

  fix $q_{i\bar{l}} \leftarrow q^c_{i\bar{l}} \quad \bar{l} \neq l, i \in I^{RM}$

  for $j \in \{j \mid \bar{w}_{lj} = 0\}$ **do**

    fix $x_{ij} \leftarrow \bar{x}^c_{ij} \quad \forall i \in I^{RM}$

    fix $v_{i\bar{l}j} \leftarrow v^c_{i\bar{l}j} \quad \forall i \in I^{RM}, \bar{l} \neq l$

  end

  solve **FindCloseLin** $\rightarrow x^\star, q^\star, v^\star, (y^s)^\star$

  update $x^c \leftarrow x^\star, v^c \leftarrow v^\star, q^c \leftarrow q^\star$

  $d_{lj} \leftarrow (y^U_{lj} - y^L_{lj})/4$

  update $y^L_{lj} \leftarrow (y^s_{lj})^\star - d_{lj}$, $y^U_{lj} \leftarrow (y^s_{lj})^\star + d_{lj}$

  if $y^L_{lj} < \bar{y}^L_{lj}$ **then** $y^L_{lj} \leftarrow \bar{y}^L_{lj}$

  if $y^U_{lj} > \bar{y}^U_{lj}$ **then** $y^U_{lj} \leftarrow \bar{y}^U_{lj}$

  solve **FindCloseLin** $\rightarrow x^\star, q^\star, v^\star, (y^s)^\star$

  solve **FindCloseNonLin** $\rightarrow x^\star, q^\star, v^\star, (y^s)^\star$

end

update $\bar{x} \leftarrow x^\star, \bar{q} \leftarrow q^\star, \bar{v} \leftarrow v^\star$

Algorithm 1: Correction of Initial Solution

2.4.4.1 Initial Point for the Approximations

The approximations described in section 2.4.3 are dependent on the point chosen for computing the first order approximation, $\tilde{x}$. By approximating a cubic polynomial by a linear function the approximation will only be valid in a small neighborhood around the point where the approximation is computed. To define the initial point for the problem we first

define the set $I^K \subseteq I^{RM}$ to be the set of indexes for which there exists at least one polynomial corresponding to a property constraint $P_{jk}(\cdot)$ with $k \in T_S$ that has a nonzero term involving x_{ij}. A more rigorous definition of I^K would be to note that a polynomial $p(x)$ of degree m in n variables can also be written in the form

$$p(x) = \sum_{|\alpha| \leq m} a_\alpha x^\alpha \tag{31}$$

where in Equation 31 each index α is an $n-$dimensional vector of non-negative integers, $|\alpha| = \sum_i \alpha_i$, and $x^\alpha = \prod_{\{i|\alpha_i \neq 0\}} x_i^{\alpha_i}$ (with the convention that $x^\alpha = 1$ for $\alpha = \mathbf{0}$). With this definition of a polynomial the set I^K can be defined as in Equation 32

$$I^K = \{i \in I^{RM} \mid \exists j, k \in T_C \text{ such that } P_{jk}(x) = \sum_{|\alpha| \leq 3} a_\alpha x^\alpha \text{ and } a_\alpha \neq 0 \text{ for } \alpha \text{ with } \alpha_i > 0\} \tag{32}$$

The initial point for the problem is chosen in the following way:

$$x_{ij} = \begin{cases} \bar{x}_{ij} & i \in I^{RM} \setminus I^K \\ X_{ij} & i \in I^K \end{cases} \tag{33}$$

where in Equation 33 $\bar{x}_{ij}$ is the corrected solution from section 2.4.4 and $X_{ij} \sim U(x_{ij}^l, x_{ij}^u)$.

2.4.5 Heuristics to Find Solutions

We describe the main heuristic used to find feasible solutions. Given the approximations to the original constraints of problem **MaxAvg** just described in sections 2.4.2 and 2.4.3 we construct an approximation to this problem, denoted by **MaxAvgLinAP** and defined in Equation 34

$$
\begin{aligned}
\text{Maximize} \quad & \sum_{j \in FP, k \in T_C} \omega_{jk} \tilde{P}_{jk}(x_{\cdot j}; \tilde{x}) & & \\
\text{Subject to :} \quad & \sum_{i \in I^{RM}} x_{ij} & = 1 \quad & \forall j \in FP \\
& \sum_{i \in I^{RM}} q_{il} & = 1 \quad & \forall l \in I^P \\
& \sum_{l \in I^P} w_{lj} & = 1 \quad & \forall j \in FP \\
& \sum_{l \in I^P} v_{ilj} + z_{ij} & = x_{ij} \quad & \forall i \in I^I, j \in FP \\
& y_{lj} & \leq w_{lj} \quad & \forall l \in I^P, j \in FP \\
& T_i^L \leq \sum_{j \in FP} D_j x_{ij} & \leq T_i^U \quad & \forall i \in I^{RM} \\
& \sum_{i \in I^{RM}} v_{ilj} & = y_{lj} \quad & \forall l \in I^P, j \in FP \\
& \tilde{P}_{jk}(x_{\cdot j}; \tilde{x}) & \geq \bar{P}_{jk} \quad & \forall j \in FP, k \in T_S \\
& q_{il} \cdot \bar{y}_{lj}^s - (1 - y_{lj}^s) & \leq v_{ilj}, \quad & s \in S_{lj}, l \in I^P, j \in FP \\
& q_{il} \cdot \bar{y}_{lj}^s + (1 - y_{lj}^s) & \geq v_{ilj}, \quad & s \in S_{lj}, l \in I^P, j \in FP \\
& \sum_{s \in S_{lj}} y_{lj}^s & = w_{lj}, \quad & l \in P, j \in FP \\
& \sum_{j \in FP} D_j \sum_{i \in I^{RM}} c_i \cdot x_{ij} & \leq \bar{c} & \\
& \bar{c}_j^L \leq D_j \sum_{i \in I^{RM}} c_i \cdot x_{ij} & \leq \bar{c}_i^U \quad & \forall j \in FP \\
& x \geq 0, \; q \geq 0, \; v \geq 0, \; w \text{ binary} & &
\end{aligned}
\tag{34}
$$

We assume that we begin with a corrected benchmark solution as given by Algorithm 1. The main heuristic to find feasible solutions is described in Algorithm 2.

input : Initial Solution: $\bar{x}^0, \bar{v}, \bar{q}$

output: Heuristic Solution: $x^\star, v^\star, q^\star, w^\star$

repeat

  get initial point $\tilde{x}$ from procedure in Section 2.4.4.1

  $\bar{y}^s_{lj} \leftarrow y^L_{lj} + (s-1)(y^U_{lj} - y^L_{lj})/(|S_{lj}| - 1)$

  $\bar{P}_{jk} \leftarrow P_{jk}(\bar{x})$

  solve **MaxAvgLinAP** $\rightarrow x^\star, q^\star, v^\star, (y^s)^\star, y^\star, z^\star, w^\star$

  update $\tilde{P}_{jk}(\cdot; x^\star)$ *around* $x^\star$

  solve **MaxAvgLinAP** $\rightarrow x^\star, q^\star, v^\star, (y^s)^\star, y^\star, z^\star, w^\star$

  solve **MaxAvg** $\rightarrow x^\star, q^\star, v^\star, (y^s)^\star, y^\star, z^\star, w^\star$

until *Solution is Feasible*

Algorithm 2: Heuristic to Find Feasible Solutions

Usually the procedure finds feasible solutions in the first passage but there might be runs where the procedure needs to be ran more than once. This is the reason for wrapping the *repeat-until* loop.

2.4.6 Improving Existing Feasible Solutions

Given a feasible solution as found in Algorithm 2 from section 2.4.5, we have no guarantee that this solution is globally optimal. In fact our focus is to be able to find these feasible solutions quickly and then choose a subset of these that are good candidates to be implemented in practice. The overall quality of the solutions is usually good but the process finds a solution that is only locally optimal and the problem is highly non-convex so we can use these solutions as a starting point to find improved solutions. The improvement considered here is simply a better objective function for the considered model (**MaxAvg**), but another possibility would be to reduce the overall cost of the solution while keeping the current levels of the qualities.

The overall idea for this improvement stage is to fix the assignments from pools to final products as in the provided solution and then decompose the problem per pool. Each problem associated with each pool is then solved in such a way that the overall solution remains feasible. To try to improve the objective function of the problem associated with

a specific pool we consider all the raw materials that are associated with the objective function of the problem being solved, I^K, as defined in Equation 32. We consider subsets of these raw materials, in this case of equal cardinality, and solve subproblems corresponding to each of these subsets. All the raw materials that are not in the current considered subset will be fixed at the current values and the resulting subproblem solved to global optimality. This is summarized in algorithm 3.

Input: $\bar{x}, \bar{q}, \bar{v}, \bar{w}$ (feasible solution), $MaxIter, MinImprov$

set $x^c \leftarrow \bar{x}, q^c \leftarrow \bar{q}, v^c \leftarrow \bar{v}, y^c \leftarrow \bar{y}$

fix $w_{lj} \leftarrow \bar{w}_{lj}, l \in I^P, j \in FP$

set $S(I^K) \leftarrow \{A_\alpha \mid A_\alpha \subseteq I^K, |A_{\alpha_1}| \approx |A_{\alpha_2}|\}$

set $nIter \leftarrow 0$

repeat

- *set* $totalImprov \leftarrow 0$
- *set* $nIter \leftarrow nIter + 1$
- **for** $l \in I^P$ **do**
  - *fix* $x_{ij} \leftarrow x^c_{ij}, i \in I^{RM}, j \in \{j \in FP \mid w_{lj} = 0\}$
  - **for** $A \in S(I^K)$ **do**
    - *fix* $x_{ij} \leftarrow x^c_{ij}, i \notin A, j \in \{j \in FP \mid w_{lj} = 1\}$
    - *set* $oldObj \leftarrow$ (value of current solution)
    - *solve* **MaxAvg** *globally* $\rightarrow x^c, q^c, v^c, y^c, z^c$
    - *set* $newObj \leftarrow$ (value of current solution)
    - *set* $totalImprov \leftarrow totalImprov + (newObj - oldObj)$
    - *unfix* $x_{ij}, i \notin A, j \in \{j \in FP \mid w_{lj} = 1\}$
  - **end**
  - *unfix* $x_{ij}, i \in I^{RM}, j \in \{j \in FP \mid w_{lj} = 0\}$
- **end**

until $(totalImprov \leq MinImprov)$ *or* $(nIter > maxIter)$

Algorithm 3: Improving Feasible Solutions

In Algorithm 3 the choice of $S(I^K)$ is quite important. If the subsets are to large the

corresponding subproblems become hard to solve (as an extreme example taking $S(I^K) = I^K$ makes each of subproblems to be solved correspond to each pool, which is a very difficult problem to be solved by itself), while choosing very small subsets would make the problem easier to solve, but the solution would not provide enough improvement (again an extreme example would be to take a partition with subsets of cardinality one). In our case we define $S(I^K)$ to be the set of subsets of I^K with cardinality two, that is:

$$S(I^K) = \{\{i_1, i_2\} \mid i_1, i_2 \in I^K, i_1 \neq i_2\} \tag{35}$$

In the general case having even having chosen a particular definition for the set $S(I^K)$, several issues would be important to take into account:

- Do we really need all the subsets in the definition of $S(I^K)$?
- What is a good order to consider for the subsets in $S(I^K)$, such that if the subproblems are solved in that order, good improvement is obtained in the initial iterations of the algorithm but not necessarily after that?

We leave these issues as future research.

2.4.7 Obtaining Nearly Feasible Solutions for AllConstrained

Up to this point we have not discussed any method to obtain solutions for model **AllConstrained**. Recall that the main difference between this model and model **MaxAvg** is the fact that in model **AllConstrained** we seek to minimize the overall cost of the portfolio while keeping explicit bounds on the property functions, while in model **MaxAvg** we have explicit constraints on cost and maximize a weighted function of some of the property functions. In order to obtain solutions to model **AllConstrained** we assume that the following property holds for any of the solvers used to solve any of the resulting subproblems: *If a feasible solution is provided as an initial solution then the final solution will be a feasible solution of at least as good value.* We start with a feasible solution for problem **MaxAvg**. At a given iteration with current solution x^c we check which of the desired property levels are met, the ones that are already met will be part of the problem as constraints the ones

that are not met will continue to be part of the objective function. In the end of the procedure if all constraints are met we can then use this point as a starting point for the problem **AllConstrained**. To describe the procedure in more detail we first define a new model **MaxAvgMod** which has a similar objective function as the model **MaxAvg**, except that only the property functions that are not satisfied for the initial solution $\bar{x}$ are part of it, and the ones that are added to the model as explicit constraints. Model **MaxAvgMod** is listed explicitly in Equation 36

$$
\begin{aligned}
\text{Maximize} \quad & \sum_{(j,k)\in(FP\times T_C)\setminus T_C^G} \omega_{jk} P_{jk}(x_{\cdot j}) \\
\text{Subject to :} \quad & \sum_{i\in I^{RM}} x_{ij} = 1 && \forall j \in FP \\
& \sum_{i\in I^{RM}} q_{il} = 1 && \forall l \in I^P \\
& \sum_{l\in I^P} w_{lj} = 1 && \forall j \in FP \\
& \sum_{l\in I^P} v_{ilj} + z_{ij} = x_{ij} && \forall i \in I^I, j \in FP \\
& y_{lj} \le w_{lj} && \forall l \in I^P, j \in FP \\
& T_i^L \le \sum_{j\in FP} D_j x_{ij} \le T_i^U && \forall i \in I^{RM} \\
& \sum_{i\in I^{RM}} v_{ilj} = y_{lj} && \forall l \in I^P, j \in FP \\
& P_{jk}(x_{\cdot j}) \ge \bar{P}_{jk} && \forall j \in FP, k \in T_S \\
& P_{jk}(x_{\cdot j}) \ge \bar{P}_{jk} && \forall (j,k) \in T_C^G \\
& q_{il} \cdot y_{lj} = v_{ilj} && \forall i \in I^{RM}, l \in I^P, j \in FP \\
& x \ge 0,\ q \ge 0,\ v \ge 0,\ w \text{ binary}
\end{aligned}
\tag{36}
$$

In Equation 36 T_C^G is defined as

$$
T_C^G = \left\{ (j,k) \in FP \times T_C \mid P_{jk}(\bar{x}_{\cdot j}) \ge \bar{P}_{jk} \right\} \tag{37}
$$

which is the set of indexes for which the initial solution satisfies the property constraints.

The overall procedure is summarized in Algorithm 4

Input: $\bar{x}, \bar{q}, \bar{v}, \bar{w}$ (feasible solution), $MaxIter$

set $x^c \leftarrow \bar{x}, q^c \leftarrow \bar{q}, v^c \leftarrow \bar{v}, y^c \leftarrow \bar{y}$

set $nIter \leftarrow 0$

set $T_C^G \leftarrow \left\{(j,k) \in FP \times T_C \mid P_{jk}(\bar{x}_{\cdot j}) \geq \bar{P}_{jk}\right\}$

while $T_C^G \neq \emptyset$ *and* $(nIter \leq maxIter)$ **do**

  set $T_C^{G'} \leftarrow T_C^G$

  solve **MaxAvgMod** $\rightarrow x^c, q^c, v^c, y^c, z^c$

  set $T_C^G \leftarrow \left\{(j,k) \in FP \times T_C \mid P_{jk}(x^c_{\cdot j}) \geq \bar{P}_{jk}\right\}$

  set $nIter \leftarrow nIter + 1$

  if $T_C^{G'} = T_C^G$ **then** break

end

set $\bar{P}_{jk} \leftarrow P_{jk}(x^c_{\cdot j})$

solve **AllConstrained** $\rightarrow x^\star, q^\star, v^\star, y^\star, z^\star, w^\star$

Algorithm 4: Obtaining Nearly Feasible Solutions for AllConstrained

2.5 Computational Results

2.5.1 Problem Dimensions and Parameters

The main parameters for the dimension of the problem we solve are shown in Table 2. The

Table 2: Dimensions of Problem

N_{RM}	N_P	N_{FP}	N_A
40	4	10	32

variables $x_{ij}, y_{lj}, z_{ij}, q_{il}$ are bounded in $[0,1]$ since they are fractions but for some of the variables/indexes we have available tighter bounds (e.g. we know that for the variables y_{lj}, the fraction coming from pool l to output j, its value must be at least 0.65; this is a constraint from the industrial process in our example). The number of pooling constraints in the problem, as defined by Equation 9 is $\left|\{i \in I^{RM} \mid i \text{ is in pool}\}\right| \cdot \left|I^P\right| \cdot |FP| = 28 \times 4 \times 10 = 1120$. Regarding which variables we choose to be partitioned our specific case we note that we have $28 \times 4 = 112$ q_{il} variables and $4 \times 10 = 40$ y_{lj} variables so we choose the variables y_{lj} to be the ones to be partitioned.

We implemented the mathematical formulations previously described in the modeling language **GAMS** (*General Algebraic Modeling System*, [1], [9], and [61]), version 23.8.1. With **GAMS** optimization models can be built in a compact form; the main components of the language that allow this are sets, variables, constraints, and data. A model in **GAMS** is defined by groups of constraints and possible other models. Constraints are defined as relations between the data and the variables. When the constraints are defined the data must be declared (the symbol corresponding to the data must exist), but might not be defined yet. So changing some of the data allows us to define different instances of on optimization problem. The main advantage of using **GAMS** is that once a model is defined we can solve it using several solvers specialized for that specific type of model, as long as they are part of the set of solvers provided by **GAMS**. The **GAMS** solvers used in this study are shown in Table 3. **DICOPT** is a solver for MINLP (*Mixed Integer Nonlinear Programs*)

Table 3: Gams Solvers Used

Problem Type	Solver
MINLP	DICOPT BARON
MILP	CPLEX
NLP	CONOPT SNOPT IPOPT

developed by J. Viswanathan and Ignacio E. Grossmann [7]. The main ideas in this solver are *Outer Approximation*, *Equality Relaxation*, and *Augmented Penalty*. NLP relaxations are solved, in some iterations taking all the variables as continuous in others the integer variables will be fixed at some values. Outer approximations are added to the model. These outer approximations are valid if the problem is convex but may cut the optimal solution in the case of non-convex problems. **BARON** (*Branch And Reduce Optimization Navigation*) is also a solver for MINLP problems, developed by Nick Sahinidis and Mohit Tawarmalani ([78] and [74]). While **DICOPT** requires the problem to be convex in order to guarantee global optimality of the solution, **BARON**, under mild assumptions such as availability of finite bounds on the variables and expressions in the problem, will provide a global optimal solution to the problem. The algorithms implemented in **BARON** are of the branch-and-bound type enhanced with constraint propagation and duality techniques which allow the

reduction of the range of the variables throughout the algorithm. Due to the fact that it aims to provide strong guarantee of global optimality **BARON** can be quite slow in general problems. To solve **MILP** (*Mixed Integer Linear Programs*) we use **CPLEX**, originally developed by Robert E. Bixby ([6], [27]). The three **NLP** (*Nonlinear Problems*) are used **CONOPT** ([33], [5]), **SNOPT** ([41], [8]), and **IPOPT** ([4]). The downsides of choosing **GAMS** as the language to implement a model is that this modeling language does not possess the flexibility and expressiveness of other computer languages. This makes it very hard to design algorithms with general data structures or perform text manipulation on the data.

2.5.2 Results of Primal Heuristic

We show results for the heuristic described in Algorithm 2. We run 10 different instances of the algorithm. The number of points in the discretization of the pooling constraints when solving the linear approximation problem, **MaxAvgLinAP**, is 10 for all instances. The solution of the second **MIP** problem is saved and used as the starting point for the problem **MaxAvg**, solved using **DICOPT**. We attempted to solve this problem using **BARON** as the **MINLP** solver but could get significant progress after a day of computation. We test 3 different **NLP** solvers in **DICOPT** to assess the performance and quality of the solution obtained using different solvers. The solvers tested were **SNOPT**, **IPOPT**, and **CONOPT**. The objective function values shown in Tables 4 and 5 are computed relatively to an existing benchmark solution. So the value -0.686 as the *MIP Obj Val* means that the objective function value of the **MIP** problem was 0.686 below the objective function of the benchmark solution.

Analyzing the results we see that the best solution obtained does not depend strongly on the value of the **MIP** solution. We also see that the computational times are not significant. The **NLP** solver that seems to deliver the best solutions is **IPOPT** at the expense of a large computational time. **CONOPT** was not the fastest in any of the reported instances but it is on average the fastest. **SNOPT** seems to deliver good solutions and it is usually almost as fast as **CONOPT**. **IPOPT** is clearly the solver that gives the solutions with

Table 4: Best Objective Function per Instance

Instance	MIP Obj Val	NLP Obj Val	Solver	Time (Seconds)
1	-0.686	2.145	SNOPT	14.31
2	-0.879	2.506	IPOPT	326.06
3	-0.810	2.552	IPOPT	203.98
4	-0.646	2.486	SNOPT	15.58
5	-0.480	2.520	IPOPT	243.49
6	-1.100	2.159	SNOPT	8.76
7	-0.422	2.519	IPOPT	160.71
8	0.010	2.552	IPOPT	162.47
9	-1.282	2.554	IPOPT	209.71
10	-0.312	2.492	IPOPT	208.90

Table 5: Average Results Over 10 Instances for Primal Heuristic

		Min	Max	Avg
Time (Seconds)	IPOPT	156.925	326.058	211.163
	CONOPT	8.214	23.482	15.882
	SNOPT	8.494	36.902	16.754
Obj Val	IPOPT	2.027	2.554	2.417
	CONOPT	0.854	2.505	2.067
	SNOPT	1.709	2.486	2.158

the largest objective function value but it takes on average more than 10 times to solve the problem than the second fastest solver on average.

Carefully specifying a good set of options for the solvers may have a significant impact in the solver performance. Table 6 shows the effects of using non-default options to solve problem **MaxAvgLinAP**. In this case the number of points used to discretize one of the variables in the pooling equations was 30, and the results are taking over 5 different runs. We specify the settings for **CPLEX** solver in Table 7. The default value of the options, if

Table 6: Effect of **CPLEX** Option File

	Without Settings File	With Setting File
Min	8.69	3.64
Max	20.53	4.18
Avg	12.634	3.826

non-zero, is given in parentheses following the value we use. In general leaving the setting at its default value instructs **CPLEX** to automatically adjust this option.

Table 7: CPLEX Option File Settings

aggcutlim=1 (3)	aggrigation limit for cut generation
brdir=1	set branching direction (Up branch selected first)
cliques=3	clique cut generation (Aggressively)
coeredind=2 (-1)	coefficient reduction on/off
covers=2	cover cut generation (Aggressively)
cutpass=2	maximum number of cutting plane passes
cuts=5	default cut generation (Aggressively)
disjcuts=1	disjunctive cuts generation (Moderately)
divetype=3	MIP dive strategy (Guided dive)
flowpaths=1	flow path cut generation (Moderately)
fpheur=1	feasibility pump heuristic (Emphasis: feasible solution)
fraccuts=2	Gomory fractional cut generation (Aggressively)
gubcovers=2	GUB cover cut generation (Aggressively)
heurfreq=50	heuristic frequency
mipemphasis=1	MIP solution tactics (Emphasize feasibility over optimality)
mipstart=1	use MIP starting values (Yes)
mircuts=2	mixed integer rounding cut generation (Aggressively)
nodesel=3 (1)	node selection strategy (Alternate best-estimate search)
ppriind=4	primal simplex pricing (Full pricing)
preslvnd=1	node presolve selector (Force node presolve)
probe=-1	perform probing before solving a MIP (No)
startalg=1	MIP starting algorithm (Primal simplex)
subalg=1	algorithm for subproblems (Primal simplex)
symmetry=5 (-1)	symmetry breaking cuts (Extremely aggressive)
varsel=4 (10)	variable selection strategy at each node (Pseudo reduced cost based)
zerohalfcuts=1	zero-half cuts (Moderately)

2.5.3 Results For Improvement of Feasible Solutions

In order to test the heuristic presented in Algorithm 3 we first run the heuristic to find primal feasible solutions, and starting the optimal solution obtained we perform the described steps. We take as subsets to be considered all subsets of two elements of the raw materials that show up in the objective function. The subproblems are solved using **BARON** which guarantees global optimality of the problems within the specified absolute and relative tolerances but has increases the running time of the overall procedure. In Table 8 we show the summary of 10 instances of the algorithm, and in Tables 9 and 10 we pick one of the instances (in this case the 5-th instance) and show, for the iterations where the improvement is not negligible the value of the improvement with respect to the starting solution. We can conclude that in general this approach does improve the initial solution but we can also

Table 8: Improvement for 2-Pair Selection

	First Round		Second Round		
	NLP	Bench	NLP	Bench	Total Time
1	0.69	2.03	0.72	2.06	859
2	0.80	2.50	0.82	2.52	1130
3	0.39	2.26	0.45	2.32	1358
4	2.31	0.57	2.31	0.57	1927
5	0.95	2.77	1.03	2.85	1145
6	0.54	2.50	0.55	2.51	1024
7	1.15	3.14	1.19	3.18	523
8	0.42	1.60	0.43	1.61	921
9	0.64	2.64	0.65	2.66	943
10	2.24	-1.46	2.31	-1.39	621
Min	0.39	-1.46	0.43	-1.39	523.00
Avg	1.01	1.86	1.05	1.89	1045.10
Max	2.31	3.14	2.31	3.18	1927.00

Table 9: Example of Non-Zero Iterations (Instance 5, First Passage)

Iteration	Increase	Iteration	Increase
1	2.88E-01	96	4.00E-04
2	3.76E-02	97	6.42E-03
3	1.91E-02	102	2.76E-06
4	3.27E-02	104	5.09E-06
12	1.55E-02	105	2.64E-08
28	1.80E-04	109	3.03E-08
33	5.99E-03	136	1.05E-01
49	5.08E-02	141	4.08E-04
70	4.88E-02	142	5.15E-03
71	5.47E-02	145	1.06E-05
73	4.90E-03	149	2.99E-05
76	3.49E-03	150	1.70E-07
78	1.32E-02	154	8.57E-03
82	1.16E-03	156	3.99E-05
91	1.46E-01	157	1.41E-05
93	9.53E-02	160	3.87E-07
94	0.010657	162	3.84E-08
95	7.41E-05		

Table 10: Example of Non-Zero Iterations (Instance 5, Second Passage)

Iteration	Increase
1	0.026376
33	2.56E-05
46	0.000161
48	0.01864
49	0.012826
52	0.012913
57	6.14E-07
60	0.00169
91	3.83E-05
96	8.37E-08
97	1.29E-07
136	1.73E-05
139	3.62E-08
165	1.23E-07

see that the final solution depends on the value of the solution we start from. It can also be seen that the first round of this procedure produces the biggest increase (in one of the cases no significant improvement is obtained in the second round). A nice property of this heuristic is that since we essentially do the improvement by looking at a pool a time and fix the raw materials not associated with the pool being considered, we could solve each of the subproblems associated with each pool separately and the resulting solution would be a valid one.

2.6 Conclusions for Numerical Results

We can see that the techniques developed enable us to find good solutions quickly. By choosing different starting points we have a efficient way of producing good quality feasible solutions that could be later filtered according to other criteria. Specialized treatment is given to the pooling constraints since these must be strictly satisfied in a final solution, while for some of the other constraint we may be able cope with having a small infeasibility. Given a specific solution we provide a local heuristic that improves it. Finally we show how starting from a feasible solution for problem **MaxAvg**, we can, iteratively, "guide" this solution to be closer to a feasible solution of problem **AllConstrained**.

CHAPTER III

COMPUTING UPPER BOUNDS

3.1 Computing Upper Bounds

In order to evaluate the quality of the heuristic solutions for problem **MaxAvg** from Equation 3 are, we present in this section upper bounds for this problem computed using polyhedral techniques. To build the relaxations each non-linear term in the formulation is replaced with an additional variable and the constraints corresponding to the relaxation of the non-linear term are added to the model defining the relaxation. We first define a straightforward model in which the relaxations for each non-linear term depend mainly on the lower and upper bounds of the variables involved in the term. After that we refine this approximation by considering relaxations where some of the variables in the formulation are partitioned and the corresponding upper bounds depend on the number of elements in the partitions.

We define here before presenting the models what we mean by relaxation. Given a problem **A** defined by $\min_x\{f^1(x) \mid g_i^1 \leq 0, i = 1, \ldots, m_1\}$, and a problem **B** defined by $\min_x\{f^2(x) \mid g_i^2 \leq 0, i = 1, \ldots, m_2\}$, we say problem **B** is a relaxation for problem **A** if for any x such that $g_i^1(x) \leq 0, i = 1, \ldots m_1$ we have

1. $g_i^2(x) \leq 0, i = 1, \ldots m_2$
2. $f^2(x) \leq f^1(x)$.

Note that in particular $\{x \mid g_i^1 \leq 0, i = 1, \ldots, m_1\} \subseteq \{x \mid g_i^2 \leq 0, i = 1, \ldots, m_2\}$, that is, any feasible point for problem **A** is also feasible for problem **B**; and $\min_x\{f^2(x) \mid g_i^2 \leq 0, i = 1, \ldots, m_2\} \leq \min_x\{f^1(x) \mid g_i^1 \leq 0, i = 1, \ldots, m_1\}$. When referring to a particular constraint, we say that L is a relaxation of g if, assuming that $g(x) \leq 0$ is the form of the constraint, we have $L(x, y) \leq g(x)$ for any x such that $g(x) \leq 0$ and for some y. In our particular case the constraints in the model are of the generic form

$$g(x) = a_0 + \sum_i a_i x_i + \sum_s b_s Q_s(x) \leq 0 \tag{38}$$

where $Q_s(x)$ is either a quadratic or cubic monomial on $x_1, x_2, \ldots, x_n$. The relaxation of this constraint is then of the form:

$$\begin{aligned} g^R(x) = &\quad a_0 + \textstyle\sum_i a_i x_i + \sum_s b_s Q_s^R \leq 0 \\ Q_{s,i}^L(x) \leq &\quad Q_s^R,\ \forall s \in \{s \mid b_s > 0\}, \forall i \\ Q_{s,i}^U(x) \geq &\quad Q_s^R,\ \forall s \in \{s \mid b_s < 0\}, \forall i \end{aligned} \tag{39}$$

where in Equation 38 each function $Q_{s,i}^{R^L}(x)$ and $Q_{s,i}^{R^U}(x)$ satisfy

$$Q_{s,i}^L(x) \leq Q_s^R(x) \leq Q_{s,i}^U(x) \tag{40}$$

that is, each function $Q_{s,i}^L(x)$ is a lower bound of $Q_s(x)$, and each function $Q_{s,i}^U(x)$ is an upper bound of $Q_s(x)$ in a suitable domain (not necessarily the whole domain where the function g is defined). As mentioned before when building the relaxation we add each of the constraints that define the relaxation for each individual term to the model, but the relaxation of constraint g in Equation 38 is defined by the RHS (*Right Hand Side*) of Equation 41

$$g^R(x) = a_0 + \sum_i a_i x_i + \sum_{\{s|b_s>0\}} \max_i \{Q_{s,i}^L(x)\} + \sum_{\{s|b_s<0\}} \min_i \{Q_{s,i}^U(x)\} \tag{41}$$

In the next sections for each non-linear term we list the functions $Q_{s,i}^L(x)$ and $Q_{s,i}^U(x)$ defining the lower and upper bound relaxations of each non-linear $Q_s(x)$ term in the model.

3.1.1 Simple Relaxations from McCormick Envelopes

The nonlinear terms possible in a general cubic model are xy, x^2, xyz, x^2y, and x^3. We list now the constraints defining a relaxation for these different terms existing in the model. For xy we have that the lower bounds are given by

$$\begin{aligned} xy &\geq y^l x + x^l y - x^l y^l \\ xy &\geq y^u x + x^u y - x^u y^u \end{aligned} \tag{42}$$

and the upper bounds are given by

$$\begin{aligned} xy &\leq y^l x + x^u y - x^u y^l \\ xy &\leq y^u x + x^l y - x^l y^u \end{aligned} \tag{43}$$

For the term x^2 we first note that this is a convex term and therefore each subgradient inequality at a given point $\bar{x}$ will constitute a valid lower bound for the term:

$$f(x) \geq f(\bar{x}) + \partial f(\bar{x})(x - \bar{x}) \tag{44}$$

Choosing more points and adding the subgradient inequalities corresponding to each of the points will improve the quality of the lower bound corresponding to each of these terms. Assuming we are going to add to the model N points, $\bar{x}^1, \bar{x}^2, \ldots, \bar{x}^N$ then these points are defined in the following way in terms of the lower and upper bounds of x, x^l and x^u:

$$\bar{x}^i = x^l + i(x^u - x^l)/(N+1), \quad i = 1, 2, \ldots, N \tag{45}$$

Using this definition the maximum distance of a point x in the interval $[x^l, x^u]$ to any of the points $\bar{x}^1, \bar{x}^2, \ldots, \bar{x}^N$ is given by $1/(N+1)$ and it occurs at the points x^l and x^u. For any point x in the interval $[\bar{x}^1, \bar{x}^N]$ the maximum distance of x to one of the points $\bar{x}^1, \bar{x}^2, \ldots, \bar{x}^N$ is $1/(2(N+1))$. Usually the endpoints of the interval defining the variable lower and upper bounds are included as one of the points where the subgradient inequality is added to the model. If this is the case the definition of the points would be

$$\tilde{x}^i = x^l + (i-1)(x^u - x^l)/(N-1), \quad i = 1, 2, \ldots, N \tag{46}$$

and the maximum distance between any point x in the interval $[x^l, x^u]$ and the points $\tilde{x}^1, \tilde{x}^2, \ldots, \tilde{x}^N$ is given by $1/(N-1)$. The maximum error in the approximation of the term x^2 if the points are defined as in Equation 45 is $(x^u - x^l)^2/(N+1)^2$ and $(x^u - x^l)^2/(4(N-1)^2)$ if the points for the subgradients are defined as in Equation 46. We prove this in Lemma 3.

Lemma 1. *Let $f(x) = x^2$ defined for $x \geq 0$ and $L(x)$ the lower approximation for f defined by the pointwise maximum of the subgradients of f added at points a and b, with $0 <= a < b$:*

$$L(x) = \max_x \{a^2 + 2a(x-a), b^2 + 2b(x-b)\} \tag{47}$$

Then the maximum difference between $f(x)$ and $L(x)$ in the interval $[a, b]$ is given by $(b-a)^2/4$ and it is attained at the point $(a+b)/2$.

Proof. Denote $f_a(x) = a^2+2a(x-a)$, $f_b(x) = b^2+2b(x-b)$. It is easy to see that $f(x)-f_a(x)$ is increasing in $[a,b]$ and $f(x)-f_b(x)$ is decreasing in the same interval, so if there is a point $z \in [a,b]$ where $f_a(z) = f_b(z)$ that point corresponds to where the maximum difference is attained. We have $f_a(\frac{a+b}{2}) = a^2 + 2a((a+b)/2 - a) = a^2 + 2a(b-a)/2 = ab$ and $f_b(\frac{a+b}{2}) = b^2 + 2b((a+b)/2 - b) = b^2 + 2b(a-b)/2 = ab$ so $f_a(\frac{a+b}{2}) = f_b(\frac{a+b}{2})$. At this point

$$(\frac{a+b}{2})^2 - f_a(\frac{a+b}{2}) = (\frac{a+b}{2})^2 - a^2 - 2a(\frac{a+b}{2} - a) = (\frac{a+b}{2})^2 - a^2 - 2a(\frac{-a+b}{2}) = (\frac{a+b}{2})^2 - ab = (b-a)^2/4 \quad (48)$$

which depends only on the length of the interval $[a,b]$. □

Lemma 2. *Let $f(x) = x^2$ defined for $x \geq 0$ and $f_a(x) = a^2+2a(x-a)$ the function defining the subgradient inequality at point a. Then $f(b) - f_a(b) = (b-a)^2$, $\forall b$.*

Proof. We have $f(b) - f_a(b) = b^2 - (a^2 + 2a(b-a)) = b^2 - (a^2 + 2ab - 2a^2) = (b-a)^2$. □

Lemma 3. *If the points for the subgradients are chosen as in Equation 45 then the maximum error in the interval $[x^l, x^u]$ between the function $f(x) = x^2$ and the approximation $f_1(x) = \max_{i=1,\ldots,N}\{f(\bar{x}^i) + \partial f(\bar{x}^i)(x - \bar{x}^i)\}$ is given by $(x^u - x^l)^2/(N+1)^2$. In the case where the points for the subgradients are defined as in Equation 46 then the maximum error between $f(x)$ and the corresponding lower approximation function $f_2(x) = \max_{i=1,\ldots,N}\{f(\tilde{x}^i) + \partial f(\tilde{x}^i)(x - \tilde{x}^i)\}$ is given by $(x^u - x^l)^2/(4(N-1)^2)$.*

Proof. For convenience with the notation we use define the functions $f_{1,i}(x)$ and $f_{2,i}(x)$ by $f_{1,i}(x) = (\bar{x}^i)^2 + 2\bar{x}(x - \bar{x})$ and $f_{2,i}(x) = (\tilde{x}^i)^2 + 2\tilde{x}(x - \tilde{x})$, $i = 1,\ldots,N$, so that $f_1(x) = \max_i\{f_{1,i}(x)\}$ and $f_2(x) = \max_i\{f_{2,i}(x)\}$. We first handle the case where the points for the subgradients are given as in Equation 45. By partitioning the interval $[x^l, x^u]$ it is clear that that the maximum error is the maximum of the errors for each of the intervals $[x^l, \tilde{x}^1], [\tilde{x}^1, \tilde{x}^2], \ldots, [\tilde{x}^{N-1}, \tilde{x}^N], [\tilde{x}^N, x^u]$. In $[x^l, \tilde{x}^1]$ the maximum error is given by $(x^u - x^l)^2/(N+1)^2$, since in this case the error depends only on the point where the subgradient is defined, $\tilde{x}^1$, and the point where the error is computed, x^l. By Lemma 2 this value is given by $((x^l + (x^u - x^l)/(N+1)) - x^l)^2 = (x^u - x^l)^2/(N+1)^2$. The same value is

obtained for the interval $[\tilde{x}^N, x^u]$. For each of the intervals $[\tilde{x}^i, \tilde{x}^{i=1}], i = 1, \ldots, N-1$ the maximum error is given by $((x^l + (i+1)(x^u - x^l)/(N+1)) - (x^l + i(x^u - x^l)/(N+1)))^2/4 = (x^u - x^l)^2/(4(N+1)^2)$, a direct application of Lemma 1. The first part of the result now follows since $(x^u - x^l)^2/(4(N+1)^2) < (x^u - x^l)^2/(N+1)^2$. For the second part of the result we use again Lemma 1 obtaining the maximum error of $((x^l + i(x^u - x^l)/(N-1)) - (x^l + (i-1)(x^u - x^l)/(N-1)))^2/4 = (x^u - x^l)^2/(4(N-1)^2)$. □

We note that neither of the two choices mentioned here for the points where the subgradient inequality is added is optimal in minimizing the maximum error obtained between the function and the lower approximation. For completeness we show that if we want to choose N points then the optimal choice would be given by

$$z_i = x^l + (2i-1)/(2N)(x^u - x^l),\ i = 1, \ldots, N \tag{49}$$

We prove this in Lemma 5 using Lemma 4.

Lemma 4. *Given an interval $I = [l, u]$ the optimal choice for placing n points $l \leq z_1 \leq z_2 \leq \cdots \leq z_n \leq u$ such that the distance from any point $x \in I$ to one of the points z_i is minimized is given by $z_i = l + (2i-1)/(2n)(u-l),\ i = 1, \ldots, n$.*

Proof. The problem that we are trying to solve here is

$$\min_{l \leq z_1 \leq \cdots \leq z_n \leq u} \left\{ \max_{x \in [l,u], i=1,\ldots,n} |z_i - x| \right\} \tag{50}$$

The solution to the inner maximization problem is given by $\max\{z_1 - l, (z_2 - z_1)/2, (z_3 - z_2)/2, \ldots, (z_n - z_{N-1})/2, u - z_n\}$. This follows from the fact that for any x in the interval $[l, z_1]$ the maximum distance between x and z_1 is given by $z_1 - l$; for any point in an interval $[z_{i+1}, z_i]$ the maximum distance will be attained at the mid-point of the interval; and for the last interval $[z_n, u]$ it is clear that the maximum distance is attained at $x = u$ and it is

given by $u - z_n$. So the problem in Equation 50 reduces to

$$\begin{array}{lrll} \min\limits_{t,z_1,\ldots,z_n} & t & & \\ \text{subject to} & z_1 - l & \leq t & \\ & (z_{i+i} - z_i)/2 & \leq t, & i = 1,\ldots,n-1 \\ & u - z_n & \leq t & \\ & z_i - z_{i+i} & \leq 0, & i = 1,\ldots,n-1 \\ & l \leq z_i & \leq u, & i = 1,\ldots,n \end{array} \tag{51}$$

This Linear program is clearly feasible since $t = u - l, z_1 = z_2 = \cdots = z_n = l$ is a feasible solution. It is also below bounded since $l \leq z_1$ and $z_1 - l \leq t$ imply $0 \leq t$. So this problem as an optimal solution. Denote the optimal solution by $t^\star, z_i^\star, i = 1,\ldots,n$, and let, for convenience, $\delta = t^\star$.

Claim 1. There exists an optimal solution with $z_1^\star - l \leq (z_2^\star - z_1^\star)/2 \leq \cdots \leq u - z_n^\star$.

If this is not the case for a particular solution, we can define a new solution with this property: let $\delta_0 = z_1^\star - l, \delta_n = u - z_n^\star, \delta_i = (z_{i+1}^\star - z_i^\star)/2, i = 1,\ldots,n-1$ and p be a permutation of $\{1,2,\ldots,n\}$ such that $\delta_{p(0)} \leq \delta_{p(1)} \leq \cdots \leq \delta_{p(n)}$. The solution $z_i = l + \sum_{r=1}^{i} \delta_{p(r-1)}, i = 1,\ldots,n$ verifies $z_1^\star - l \leq (z_2^\star - z_1^\star)/2 \leq \cdots \leq u - z_n^\star$.

Claim 2. $0 < z_1^\star - l$ for all optimal solutions.

To show this we first show that allowing one extra point strictly decreases the value of the optimal solution. For simplicity we assume here that $l = 0$ and $u = 1$. Let $z_1,\ldots,z_n$ be an optimal solution verifying Claim 1. Let $0 < \varepsilon < 1 - z_n$ and define $z_i' = (1-\varepsilon)z_i, i = 1,\ldots,n, z_{n+1}' = 1 - \varepsilon$. We have $z_1' = (1-\varepsilon)z_1 \leq z_1$, $z_{i+1}' - z_i' = (1-\varepsilon)(z_{i+1} - z_i) \leq z_{i+1} - z_i, i = 1,\ldots,n-1$ and $z_{n+1}' - z_n' = (1-\varepsilon) - (1-\varepsilon)z_n = (1-\varepsilon)(1-z_n) < 1 - z_n$ since $\varepsilon > 0$ and $z_n < 1$ which shows that the new solution has a lower value than the previous one. So we cannot have a solution for which $z_1^\star = l$ since removing the point z_1 would correspond to a problem with $n-1$ points and value $t^\star$, but if a solution with $n-1$ points has value $t^\star$ then there must exist a solution with value strictly less than $t^\star$ if we allow one extra point, a contradiction.

Claim 3. The first 3 group of constraints in the LP problem in Equation 51 are satisfied at equality in any optimal solution.

Suppose this is not the case. Then we have a solution $z_1^\star, \ldots, z_n^\star$ with $z_1^\star - l < \delta$ and $0 < z_1^\star - l$ by Claim 2. Let $\varepsilon_1 > \varepsilon_2 > \cdots > \varepsilon_n > 0$ be numbers satisfying $\sum_{i=1}^n \varepsilon_i = \delta - (z_1^\star - l)$ and $\varepsilon_i < z_{i+1}^\star - z_i^\star, i = 1, \ldots, n-1, \varepsilon_n < u - z_n^\star$. We show that the solution $z_i' = z_i^\star + \varepsilon_i, i = 1, 2, \ldots, n,\ t' = \max_{i=1,\ldots,n-1}\{z_1' - l, (z_{i+1}' - z_i')/2, u - z_n'\}$ is feasible and verifies $t' < t^\star$ which contradicts the optimality of $t^\star$. The first three set of constraints are satisfied by the definition of t'. Also we have $z_{i+1}' - z_i' = z_{i+1}^\star + \varepsilon_{i+1} - (z_i^\star + \varepsilon_i) = z_{i+1}^\star - z_i^\star + (\varepsilon_{i+i} - \varepsilon_i) > z_{i+1}^\star - z_i^\star - \varepsilon_i > 0$ and all the z_i' obviously between l and u so the solution is feasible. Regarding optimality, $z_1' = z_1^\star + \varepsilon_1 - l < \delta, z_{i+1}' - z_i' = z_{i_1}^\star - z_i^\star + (\varepsilon_{i+1} - \varepsilon_i) < z_{i+1}^\star - z_i^\star < \delta, i = 1, \ldots, n-1$ and $u - z_n' = u - z_n^\star - \varepsilon_n < u - z_n^\star = \delta$ which shows that this solution has a lower objective value than the previous one, a contradiction.

Finally by using Claim 3 and the fact that $u - l = (z_1^\star - l) + (z_2^\star - z_1^\star) + \cdots + (z_n^\star - z_{n-1}^\star) + (u - z_n^\star) = \delta + 2\delta + \cdots + 2\delta + \delta = \delta + (n-1)2\delta + \delta = 2n\delta$ gives $\delta = (u-l)/(2n)$ and $z_i^\star = l + (2i-1)/(2n)(u-l)$. □

Lemma 5. *If we add the subgradients in the points as defined by Lemma 4 then the maximum error between $f(x) = x^2$, defined in the interval $[x^l, x^u]$, and the corresponding lower approximation is given by $(x^u - x^l)^2/(4N^2)$. This is the best possible lower bound.*

Proof. By Lemma 4 the optimal placement of the points defining the subgradients is given by $z_i = l + (2i-1)/(2n)(u-l),\ i = 1, \ldots, n$. The distance between the point z_{i+1} and z_i is $(x^u - x^l)/n$ which gives that the maximum error is $((x^u - x^l)/n)^2/4 = (x^u - x^l)^2/(4N^2)$. □

We now show some figures and tables that give an idea on how the choices where the subgradients are placed result in different possible errors. Figures 3 and 4 represent draw the function x^2 together with two different approximations obtained by two different choice if points, while Figures 5 and 6 represents the actual error function that results in the specific choice where the subgradients are added.

Table 11 shows values corresponding to the maximum error for each of the choice of points for placing the subgradients over 3 different intervals, $[0, 1], [1, 2]$, and $[1/2, 1]$ and 3 choices of number of points. A final note regarding the lower relaxation of x^2 is that, as mentioned before, for relaxing the problem we replace the non-linear term by new variables

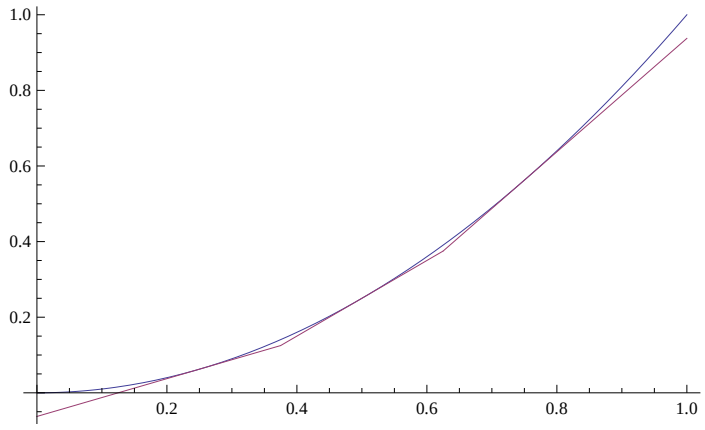

Figure 3: Approximation with our choice of points

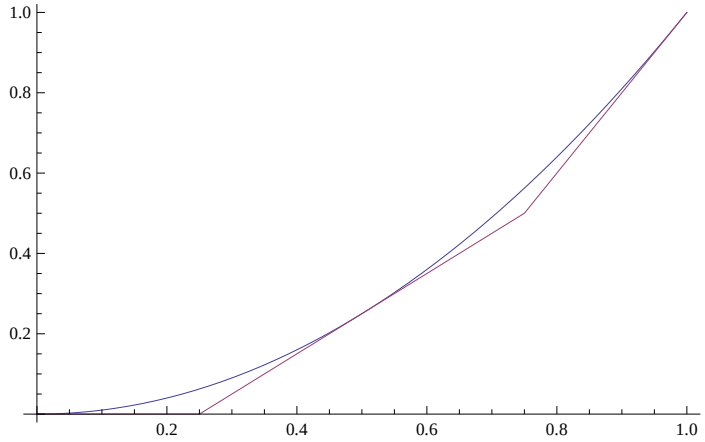

Figure 4: Approximation where lower and upper bounds are included

and add the relaxations involving the new variable. In this case since all variables are non-negative we can also define these new variables to be non-negative. In the general case when adding a new variable z replacing the non-linear term $x_i x_j$ we add the following bounds to variable z:

$$\begin{aligned} z^l = &\quad x_i^l \cdot x_j^l \\ z^u = &\quad x_i^u \cdot x_j^u \end{aligned} \tag{52}$$

This means that in particular $z_{xx} \geq (x^l)^2$ which is equivalent to adding an extra subgradient inequality if $x^l = 0$ (if $x^l > 0$ the inequality added is not as strong).

The upper relaxation for $f(x) = x^2$ in the interval $[x^l, x^u]$ is given by the secant inequality connecting the points $(x^l, (x^l)^2$ and $(x^u, (x^u)^2)$:

$$z_{xx} \leq (x^l)^2 + \frac{(x^u)^2 - (x^l)^2}{x^u - x^l}(x - x^l) = (x^u + x^l)x - x^u x^l \tag{53}$$

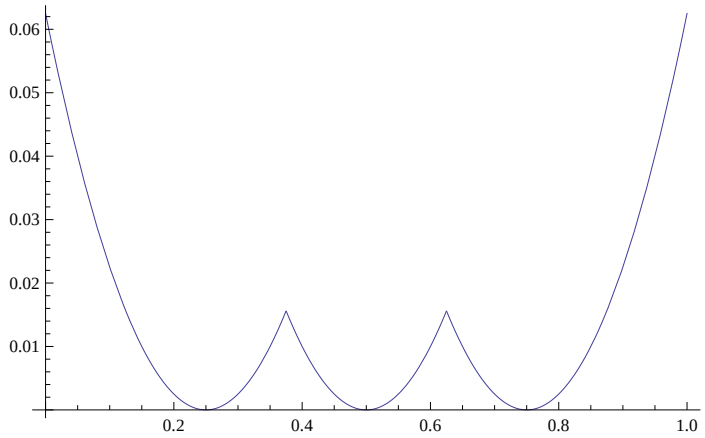

Figure 5: Error for the approximation with our choice of points

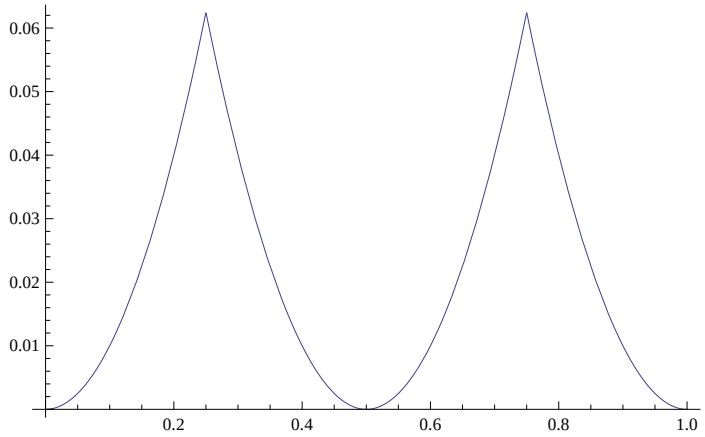

Figure 6: Error for lower approximation for x^2 where bounds are included

Table 11: Error values for different choice of points. All values divided by 100. ($\Delta = x^u - x^l$)

Interval	N. Points	$\Delta^2/(4(N-1)^2$	$\Delta^2/4N^2$	$\Delta^2/(N+1)^2$	$\Delta^2/(4(N+1)^2)$
$[0,1]$	3	6.250	2.778	6.250	1.563
	5	1.563	1.000	2.778	0.694
	7	0.694	0.510	1.563	0.391
$[1,2]$	3	6.250	2.778	6.250	1.563
	5	1.563	1.000	2.778	0.694
	7	0.694	0.510	1.563	0.391
$[1/2,1]$	3	1.563	0.694	1.563	0.391
	5	0.391	0.250	0.694	0.174
	7	0.174	0.128	0.391	0.098

It is a simple calculus exercise to show that the maximum value of the difference $((x^u + x^l)x - x^u x^l) - x^2$ in the interval $[x^l, x^l]$ is attained at the point $(x^l + x^u)/2$ and the value of the difference is given by $(\frac{x^u - x^l}{2})^2$. This relaxation is represented in Figure 7.

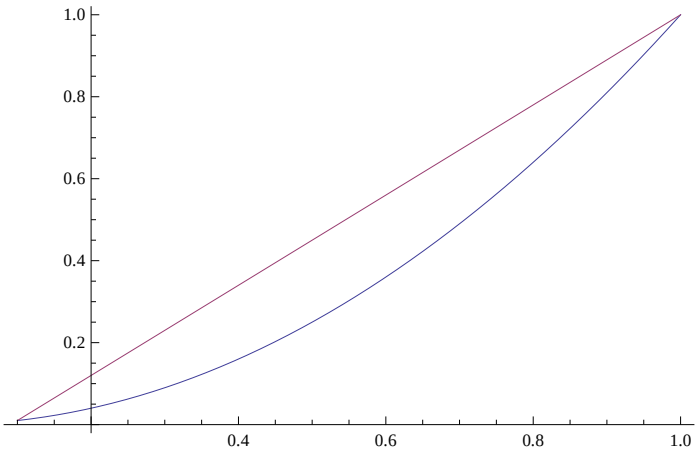

Figure 7: Secant Inequality giving upper bound for x^2 on $[x^l, x^u]$

The lower bound for x^3 is also given by adding subgradients for this function in the same points where we add subgradients for the x^2 functions:

$$\bar{x}^i = x^l + i(x^u - x^l)/(N+1), \quad i = 1, 2, \ldots, N \tag{54}$$

The constraints added to the relaxed model are in this case:

$$z_{x^3} \geq (x^i)^3 + 3(x^i)^2(x - x^i) = (x^i)^2(3x - 2x^i), \quad i = 1, \ldots, N \tag{55}$$

Lemma 6. *Consider the function $f(x) = x^3$ defined for $x \geq 0$. When considering a lower approximation to $f(x)$ defined by $l(x) = \max\{f_a(x), f_b(x)\}$ where f_a and f_b are the linear functions corresponding to the subgradients at points a and b, then the maximum difference between $f(x)$ and $l(x)$ in the interval $[a, b]$ occurs at the point $\frac{2(a^2+ab+b^2)}{3(a+b)}$ with value*

$$\frac{2(a-b)^2(2a+b)^2(a+2b)^2}{27(a+b)^3} \tag{56}$$

Proof. It is clear that the maximum difference is obtained at the intersection of the lines defined by $l_1(x) = a^3 + 3a^2(x-a)$ and $l_2(x) = b^3 + 3b^2(x-b)$ since $x^3 - l_1(x)$ is increasing in $[a, b]$ and $x^3 - l_2(x)$ is decreasing in the same interval. By solving for x and plugging the point in the expression $x^3 - l_1(x)$ we obtain the point and value desired. □

In Table 12 we give examples of the maximum error when considering possibilities for placing the subgradients. The error for "Choice 1" corresponds to choosing the points for

Table 12: Maximum Error for x^3 and Lower Approximation for Different Points/no. Points in $[0, 1]$

N. Points	Choice 1	Choice 2
3	0.156	0.137
5	0.074	0.041
7	0.043	0.019

the subgradients as in Equation 45 while for "Choice 2" corresponds to the choice of points in Equation 46.

The upper bound for the term x^3 is given by the secant equation between the points $(x^l, (x^l)^3)$ and $(x^u, (x^u)^3)$, in a way similar to the upper bound for the term x^2:

$$z_{x^3} \leq (x^l)^3 + \frac{(x^u)^3 - (x^l)^3}{x^u - x^l}(x - x^l) = -x^l x^u (x^l + x^u) + ((x^l)^2 + x^l x^u + (x^u)^2)x \tag{57}$$

By taking the derivative of $(x^l)^3 + \frac{(x^u)^3-(x^l)^3}{x^u-x^l}(x - x^l)$ we find the value where the maximum difference is attained happens at

$$\sqrt{\frac{(x^l)^2 + x^l x^u + (x^u)^2}{3}} \tag{58}$$

with value

$$(x^l)^3 - \frac{\left((x^l)^2 + x^l x^u + (x^u)^2\right)^{3/2}}{3\sqrt{3}} + \frac{1}{3}\left((x^l)^2 + x^l x^u + (x^u)^2\right)\left(-3x^l + \sqrt{3}\sqrt{(x^l)^2 + x^l x^u + (x^u)^2}\right) \tag{59}$$

Unlike the bounds obtained for x^2 the maximum value depends not only on the length of the interval but also on the points where the function is being approximated. Examples of what the maximum error is in different intervals are shown in Table 13.

Table 13: Examples of maximum error between x^3 and upper approximation in several intervals

Interval	$[0, 1]$	$[1, 2]$	$[1/2, 1]$
Error	0.3849	1.12845	0.141056

The lower bound for a term of the form x^2y is given by the constraints in Equation 60:

$$\begin{aligned} w_{x^2y} &\geq x^l(x^l(y-2y^l)+2xy^l) \\ w_{x^2y} &\geq x^u(x^l(y-2y^u)+2xy^u) \\ w_{x^2y} &\geq x^l(-x^ly^l+x^u(y-y^u)+x(y^l+y^u)) \\ w_{x^2y} &\geq x^u(x^l(y-y^l)-x^uy^u+x(y^l+y^u)) \end{aligned} \tag{60}$$

These bounds are obtained in a way similar to McCormick envelopes. We introduce the variable w_{x^2y} to replace the non-linear term x^2y where this term appears. We can introduce an intermediate variable w_{xy} to denote xy, with bounds $w^l_{xy} = x^ly^l$ and $w^u_{xy} = x^uy^u$. Then we have that the inequalities in Equation 61 are valid for w_{xy}:

$$\begin{aligned} w_{xy} &\geq y^lx + x^ly - x^ly^l \\ w_{xy} &\geq y^ux + x^uy - x^uy^u \end{aligned} \tag{61}$$

Using the fact that $x^2y = x(xy) = xw_{xy}$ we also have

$$\begin{aligned} w_{x^2y} &\geq w^l_{xy}x + x^lw_{xy} - x^lw^l_{xy} \\ w_{x^2y} &\geq w^u_{xy}x + x^uw_{xy} - x^uw^u_{xy} \end{aligned} \tag{62}$$

Recalling that the lower bounds on x and y are non-negative we can use the inequalities in Equation 61 to obtain

$$\begin{aligned} w_{x^2y} &\geq w^l_{xy}x + x^l(w^l_{xy}x + x^lw_{xy} - x^lw^l_{xy}) - x^lw^l_{xy} \\ w_{x^2y} &\geq w^l_{xy}x + x^l(w^u_{xy}x + x^uw_{xy} - x^uw^u_{xy}) - x^lw^l_{xy} \\ w_{x^2y} &\geq w^u_{xy}x + x^u(w^l_{xy}x + x^lw_{xy} - x^lw^l_{xy}) - x^uw^u_{xy} \\ w_{x^2y} &\geq w^u_{xy}x + x^u(w^u_{xy}x + x^uw_{xy} - x^uw^u_{xy}) - x^uw^u_{xy} \end{aligned} \tag{63}$$

by expanding the definitions of w^l_{xy} and w^u_{xy} we obtain the inequalities listed in Equation 60.

The inequalities defining the upper bound for the term x^2y are derived in a similar way as the inequalities for the lower bound. Starting from the inequalities

$$\begin{aligned} w_{xy} &\leq y^lx + x^uy - x^uy^l \\ w_{xy} &\leq y^ux + x^ly - x^ly^u \end{aligned} \tag{64}$$

and

$$\begin{aligned} w_{x^2y} &\leq w^u_{xy}x + x^lw_{xy} - x^lw^u_{xy} \\ w_{x^2y} &\leq w^l_{xy}x + x^uw_{xy} - x^uw^l_{xy} \end{aligned} \tag{65}$$

and replacing the inequalities in Equation 64 in the inequalities in Equation 65 we obtain the set of inequalities

$$\begin{aligned} w_{x^2y} &\leq w^u_{xy}x + x^l(y^lx + x^uy - x^uy^l) - x^lw^u_{xy} \\ w_{x^2y} &\leq w^u_{xy}x + x^l(y^ux + x^ly - x^ly^u) - x^lw^u_{xy} \\ w_{x^2y} &\leq w^l_{xy}x + x^u(y^lx + x^uy - x^uy^l) - x^uw^l_{xy} \\ w_{x^2y} &\leq w^l_{xy}x + x^u(y^ux + x^ly - x^ly^u) - x^uw^l_{xy} \end{aligned} \tag{66}$$

which after simplifying and substituting the values for the lower and upper bounds of the variable w_{xy} reduces to:

$$\begin{aligned} w_{x^2y} &\leq (x^u)^2y + (x - x^u)(x^l + x^u)y^l \\ w_{x^2y} &\leq x^l(xy^l + x^u(y - y^l - y^u)) + xx^uy^u \\ w_{x^2y} &\leq x^l(xy^l + x^u(y - y^l - y^u)) + xx^uy^u \\ w_{x^2y} &\leq (x^l)^2y + (x - x^l)(x^l + x^u)y^u \end{aligned} \tag{67}$$

One can immediately see that the second and third inequalities are identical so one of them is redundant. In fact in Chapter 4, section 4.5 we show that the two inequalities

$$\begin{aligned} w_{x^2y} &\leq (x^u)^2y + (x - x^u)(x^l + x^u)y^l \\ w_{x^2y} &\leq (x^l)^2y + (x - x^l)(x^l + x^u)y^u \end{aligned} \tag{68}$$

are actually sufficient to describe the same set as the set described by the inequalities in Equation 67, and these inequalities define the strongest concave set possible to this non-linear term (the concave envelope).

Finally for building upper and lower bounds for trilinear terms (terms of the form xyz) we use McCormick envelopes [62]. In [73] Ryoo and Sahinidis discuss four techniques for bounding general multilinear functions. The four techniques mentioned are arithmetic intervals, recursive arithmetic intervals, logarithmic transformation, and exponent transformation. Only the two first techniques are applicable in our case since they are polyhedral; the other two make use non-linear functions to bound the trilinear term. To use the logarithmic transformation it is required that the variables be strictly positive. The idea for the logarithmic transformation is that

$$xyz = \exp(\log(xyz)) = \exp(\log(x) + \log(y) + \log(z)) \tag{69}$$

and for each of the variables x, y, and z we have the following inequality

$$\log(x) \geq \log(x^u) + \frac{\log(x^u/x^l)}{x^u - x^l}(x - x^u) \tag{70}$$

where the RHS of Equation 70 is simply the secant equation passing through the points $(x^l, \log(x^l))$ and $(x^u, \log(x^u))$. By replacing each of the logarithms in Equation 69 by the linear underestimator defined in Equation 70 we obtain a convex function on the variables x, y, and z which is an underestimator of xyz. The exponent transformation relies on the fact that

$$\begin{aligned} xyz =& \frac{1}{48}\Big\{(x+y+z)^3 - (x+y-z)^3 \\ & - (x-y+x)^3 - (-x+y+z)^3 \\ & + (x-y-z)^3 + (-x+y-z)^3 \\ & + (-x-y+z)^3 - (-x-y-z)^3\Big\} \\ =& \frac{1}{24}\Big\{(x+y+z)^3 - (x+y-z)^3 \\ & - (x-y+z)^3 + (x-y-z)^3\Big\} \end{aligned} \tag{71}$$

The bounds for the trilinear term are then obtained by replacing each of the non-linear terms in Equation 71 by an additional variable and using standard techniques for bounding univariate functions (the classic reference here is McCormick's paper on factorable programming, [62]).

Arithmetic intervals are generalizations of McCormick envelopes in such a way that the resulting inequalities don't require extra variables except the ones representing the non-linear term being approximated. We start from the McCormick inequalities defining the lower bound of xy

$$\begin{aligned} w_{xy} &\geq y^l x + x^l y - x^l y^l \\ w_{xy} &\geq y^u x + x^u y - x^u y^u \end{aligned} \tag{72}$$

and the inequalities defining the upper bound of xy

$$\begin{aligned} w_{xy} &\leq y^u x + x^l y - x^l y^u \\ w_{xy} &\leq y^l x + x^u y - x^u y^l \end{aligned} \tag{73}$$

To obtain the lower bounds for xyz we start from the following valid inequalities:

$$\begin{aligned} 0 &\leq (x - x^l)(y - y^l)(z - z^l) \\ 0 &\leq (x^u - x)(y^u - y)(z - z^l) \\ 0 &\leq (x^u - x)(y - y^l)(z^u - z) \\ 0 &\leq (x - x^l)(y^u - y)(z^u - z) \end{aligned} \tag{74}$$

then expanding the RHS of each of the inequalities and replacing the trilinear term xyz by the variable w_{xyz}, we obtain the following system of inequalities:

$$\begin{aligned} w_{xyz} \geq & xyz^l + xy^l z + x^l yz - xy^l z^l - x^l yz^l - x^l y^l z + x^l y^l z^l \\ w_{xyz} \geq & xyz^l + xy^u z + x^u yz - xy^u z^l - x^u yz^l - x^u y^u z + x^u y^u z^l \\ w_{xyz} \geq & xyz^u + xy^l z + x^u yz - xy^l z^u - x^u yz^u - x^u y^l z + x^u y^l z^u \\ w_{xyz} \geq & xyz^u + xy^u z + x^l yz - xy^u z^u - y^l yz^u - x^l y^u z + x^l y^u z^u \end{aligned} \tag{75}$$

Each of the bilinear terms in this system is now replaced by the corresponding two inequalities in Equation 72 which results in $4 \times 2^3 = 32$ additional constraints for the lower bound of each trilinear term. The resulting inequalities corresponding to the first inequality in Equation 75 are:

$$\begin{aligned} w_{xyz} \geq & y^l z^l x + x^l z^l y + x^l y^l z + (-2x^l y^l z^l) \\ w_{xyz} \geq & y^l z^l x + x^l z^u y + x^l y^u z + (-x^l y^l z^l - x^l y^u z^u) \\ w_{xyz} \geq & y^l z^u x + x^l z^l y + x^u y^l z + (-x^l y^l z^l - x^u y^l z^u) \\ w_{xyz} \geq & y^l z^u x + x^l z^u y + (-x^l y^l + x^u y^l + x^l y^u)z + (-x^u y^l z^u - x^l y^u z^u) \\ w_{xyz} \geq & y^u z^l x + x^u z^l y + x^l y^l z + (-x^l y^l z^l - x^u y^u z^l) \\ w_{xyz} \geq & y^u z^l x + (-x^l z^l + x^u z^l + x^l z^u)y + x^l y^u z + (-x^u y^u z^l - x^l y^u z^u) \\ w_{xyz} \geq & (-y^l z^l + y^u z^l + y^l z^u)x + x^u z^l y + x^u y^l z + (-x^u y^u z^l - x^u y^l z^u) \\ w_{xyz} \geq & (-y^l z^l + y^u z^l + y^l z^u)x + (-x^l z^l + x^u z^l + x^l z^u)y \\ & + (-x^l y^l + x^u y^l + x^l y^u)z + (x^l y^l z^l - x^u y^u z^l - x^u y^l z^u - x^l y^u z^u) \end{aligned} \tag{76}$$

For obtaining upper bounds the procedure is similar but starting in this case from the valid

inequalities:

$$\begin{aligned}
0 \leq & (x - x^l)(y^u - y)(z - z^l) \\
0 \leq & (x - x^l)(y - y^l)(z^u - z) \\
0 \leq & (x^u - x)(y - y^l)(z - z^l) \\
0 \leq & (x^u - x)(y^u - y)(z^u - z)
\end{aligned} \tag{77}$$

Expanding these inequalities and replacing the trilinear term xyz by the variable w_{xyz} we obtain:

$$\begin{aligned}
w_{xyz} \leq & xyz^l + x^l yz + xy^u z - x^l y^u z - x^l yz^l - xy^u z^l + x^l y^u z^l \\
w_{xyz} \leq & xyz^u + x^l yz + xy^l z - x^l y^l z - x^l yz^u - xy^l z^u + x^l y^l z^u \\
w_{xyz} \leq & xyz^l + x^u yz + xy^l z - x^u y^l z - x^u yz^l - xy^l z^l + x^u y^l z^l \\
w_{xyz} \leq & xyz^u + x^u yz + xy^u z - x^u y^u z - x^u yz^u - xy^u z^u + x^u y^u z^u
\end{aligned} \tag{78}$$

To obtain the upper bound follow the same procedure as for the lower bound replacing each of the bilinear terms in the inequalities in Equation 78 by the inequalities in Equation 73. This results in additional 32 constraints added to the model. We present the constraints added to the model corresponding to the first inequality in Equation 78:

$$\begin{aligned}
w_{xyz} \leq & y^u z^u x + x^l z^u y + x^l y^l z + (-x^l y^l z^u - x^l y^u z^u) \\
w_{xyz} \leq & y^u z^u x + x^l z^l y + x^l y^u z + (-x^l y^u z^l - x^l y^u z^u) \\
w_{xyz} \leq & y^u z^l x + x^l z^u y + (x^l y^l - x^l y^u + x^u y^u) z + (-x^u y^u z^l - x^l y^l z^u) \\
w_{xyz} \leq & y^u z^l x + x^l z^l y + x^u y^u z + (-x^l y^u z^l - x^u y^u z^l) \\
w_{xyz} \leq & (y^l z^l - y^u z^l + y^u z^u) x + (-x^l z^l + x^u z^l + x^l z^u) y \\
& + x^l y^l z + (-x^u y^l z^l + x^l y^u z^l - x^l y^l z^u - x^l y^u z^u) \\
w_{xyz} \leq & (y^l z^l - y^u z^l + y^u z^u) x + x^u z^l y + x^l y^u z + (-x^u y^l z^l - x^l y^u z^u) \\
w_{xyz} \leq & y^l z^l x + (-x^l z^l + x^u z^l + x^l z^u) y \\
& + (x^l y^l - x^l y^u + x^u y^u) z + (-x^u y^l z^l + x^l y^u z^l - x^u y^u z^l - x^l y^l z^u) \\
w_{xyz} \leq & y^l z^l x + x^u z^l y + x^u y^u z + (-x^u y^l z^l - x^u y^u z^l)
\end{aligned} \tag{79}$$

Recursive arithmetic intervals differ from arithmetic intervals in the fact that when presented with the inequalities in Equations 75 and 78 instead of using the valid relaxations

to the bilinear terms to obtain relaxations in the original variables, the bilinear terms are replaced by the corresponding extra variables and the relaxations for the bilinear terms are also added to the model. The set of inequalities corresponding the lower bound obtained in this fashion for trilinear terms is shown in Equation 80.

$$\begin{aligned}
w_{xyz} \geq & w_{xy}z^l + w_{xz}y^l + x^l w_{yz} - xy^l z^l - x^l yz^l - x^l y^l z + x^l y^l z^l \\
w_{xyz} \geq & w_{xy}z^l + w_{xz}y^u + x^u w_{yz} - xy^u z^l - x^u yz^l - x^u y^u z + x^u y^u z^l \\
w_{xyz} \geq & w_{xy}z^u + w_{xz}y^l + x^u w_{yz} - xy^l z^u - x^u yz^u - x^u y^l z + x^u y^l z^u \\
w_{xyz} \geq & w_{xy}z^u + w_{xz}y^u + x^l w_{yz} - xy^u z^u - y^l yz^u - x^l y^u z + x^l y^u z^u \\
w_{xy} \geq & y^l x + x^l y - x^l y^l \\
w_{xy} \geq & y^u x + x^u y - x^u y^u \\
w_{xz} \geq & z^l x + x^l z - x^l z^l \\
w_{xz} \geq & z^u x + x^u z - x^u z^u \\
w_{yz} \geq & y^l z + z^l y - z^l y^l \\
w_{yz} \geq & y^u z + z^u y - z^u y^u
\end{aligned} \tag{80}$$

Equation 81 lists the inequalities describing the upper bound for the trilinear term in this case:

$$\begin{aligned}
w_{xyz} \leq & w_{xy}z^l + x^l w_{yz} + w_{xz}y^u - x^l y^u z - x^l yz^l - xy^u z^l + x^l y^u z^l \\
w_{xyz} \leq & w_{xy}z^u + x^l w_{yz} + w_{xz}y^l - x^l y^l z - x^l yz^u - xy^l z^u + x^l y^l z^u \\
w_{xyz} \leq & w_{xy}z^l + x^u w_{yz} + w_{xz}y^l - x^u y^l z - x^u yz^l - xy^l z^l + x^u y^l z^l \\
w_{xyz} \leq & w_{xy}z^u + x^u w_{yz} + w_{xz}y^u - x^u y^u z - x^u yz^u - xy^u z^u + x^u y^u z^u \\
w_{xy} \leq & y^u x + x^l y - x^l y^u \\
w_{xy} \leq & y^l x + x^u y - x^u y^l \\
w_{xz} \leq & z^u x + x^l z - x^l z^u \\
w_{xz} \leq & z^l x + x^u z - x^u z^l \\
w_{yz} \leq & y^u z + z^l y - z^l y^u \\
w_{yz} \leq & y^l z + z^u y - z^u y^l
\end{aligned} \tag{81}$$

The number of constraints in this case for each of the lower and upper bounds is $4+2\times3 = 10$ additional constraints per trilinear term and also 3 additional variables to represent the value of relaxation for the bilinear terms. In the general case some of these additional constraints and variables would be added to the model in any if the bilinear term shows up by itself in the model. This is the relaxation that we actually use.

3.1.2 Relaxations from Partition of Variable Domains

3.1.2.1 Background

We now present relaxations that give stronger bounds for the problem than the relaxations from the previous section at the expense of introducing new variables and constraints into the model.

The general idea for building these relaxations is that one of the variables in the non-linear term is chosen to be partitioned and it is necessary to model the requirement that the partitioned variable is in one of the partitions. There exist several ways of modeling this requirement as can be seen in [31], [54] or [67]. Assume the partition of variable x with domain $[x^l, x^l]$ is defined by

$$x \in [x^l, x^u] = \cup_{i=1,\ldots,N}[x_{i-1}, x_i] \tag{82}$$

with $x^l = x_0 < x_1 < \cdots < x_N = x^u$. One of the ways of modeling the requirement is to define binary variables $\lambda_i, i = 0, \ldots, N-1$ by $\lambda_i = 1$ if $x \in [x_i, x_{i+1}]$ and 0 otherwise and define also variables $\delta_i, i = 1, \ldots, N-1$, called a *local differential variable* corresponding to each partition. Then the requirement can be expressed by

$$\begin{aligned} x = &\sum_{i=0}^{N-1} (x_i\lambda_i + \delta_i) \\ &0 \le \delta_i \le (x_{i+1} - x_i) \cdot \lambda_i, \quad i = 0, 1, \ldots, N-1 \end{aligned} \tag{83}$$

Another possibility is that instead of having individual local differential variables $\delta_i, i = 0, \ldots, N-1$ we can have a *global differential variable*, δ, which aggregates the differential variables into a single variable ($\delta = \delta_0 + \cdots + \delta_{N-1}$). With this definition modeling the

requirement is given by

$$\begin{aligned} x = & \sum_{i=0}^{N-1} (x_i \lambda_i + \delta) \\ 0 \leq \delta \leq & \sum_{i=0}^{N-1} (x_{i+1} - x_i) \cdot \lambda_i \end{aligned} \tag{84}$$

Instead of having binary variables λ_i to enforce that x is in a given partition we can also use the following variables θ_i with

$$\begin{aligned} \theta_i = & \begin{cases} 1 & \text{if } x \geq x_{i+1} \\ 0 & \text{otherwise} \end{cases}, \quad 0 \leq i \leq N-2 \\ \theta_i \geq & \theta_{i+1}, \quad 0 \leq i \leq N-3 \end{aligned} \tag{85}$$

with this definition we define also *local incremental variables*, u_i, which we use to define x by

$$\begin{aligned} x = x^l + & \sum_{i=0}^{N-1} (x_{i+1} - x_i) u_i, \quad 0 \leq u_i \leq 1 \\ 0 \leq u_{N-1} \leq \theta_{N-2} \leq u_{N-2} \leq \theta_{N-3} \leq \cdots \leq u_1 \leq \theta_0 \leq u_0 \leq & 1 \end{aligned} \tag{86}$$

We can also use a *global incremental variable* (we use again the notation δ) in this case, giving the following representation for x:

$$\begin{aligned} x = x^l + & \textstyle\sum_{i=0}^{N-2} (x_{i+1} - x_i) \theta_i + \delta \\ 0 \leq \delta \leq & (x_2 - x_1) + \textstyle\sum_{i=0}^{N-2} (x_{i+1} - x_i) \theta_i \end{aligned} \tag{87}$$

We have the following relaxation between the variables $\lambda_i, \theta_i, \delta_i$, and u_i:

$$\begin{aligned} \lambda_0 &= 1 - \theta_0 \\ \lambda_{i+1} &= \theta_i - \theta_{i+1}, \quad i = 0, \ldots, N-3 \\ \lambda_{N-1} &= \theta_{N-1} \\ \delta_i &= (x_{i+1} - x_i) \left(u_i + \textstyle\sum_{j=0}^{i-1} \lambda_j - 1 \right) \end{aligned} \tag{88}$$

It is obvious that the formulations with the local differential variables δ_i and local incremental variables u_i introduce more constraints and variables in the problem, but since for any feasible solution feasible for these formulations we can build a feasible solution for the formulation with global differential and incremental variables the later formulations cannot be stronger than the first. Although Equation 88 shows that the two ways of defining the binary variables are closely related, Padberg in [67] shows that the two formulation have

different properties in what concerns their linear programming relaxations. He shows that the formulation with the θ_i variables is locally ideal and that the linear programming relaxation of the set defined by the θ_i variables is properly contained in the linear programming relaxation of the set defined by the λ_i variables.

Wicaksono and Karimi in [86] present several alternatives to build strong relaxations for bilinear problems based on piecewise MILP under and over estimators for the bilinear terms. The relaxations are divided in three major categories: Big-M Formulations, Convex Combination Formulations, and Incremental Cost Formulations. We present specific formulation which are representative of the formulations in each category. Define $z = x \cdot y$. The big-M formulation can be defined by

$$
\begin{aligned}
&\lambda_i = \begin{cases} 1 & \text{if } x_i \le x \le x_{i+1} \\ 0 & \text{otherwise} \end{cases}, \forall i = 0, 1, \ldots, N-1 \\
&\textstyle\sum_{j=0}^{N-1} \lambda_i = 1 \\
&x \;\; \ge x_i \cdot \lambda_i + x^l(1-\lambda_i), \quad \forall i \\
&x \;\; \le x_{i+1} \cdot \lambda_i + x^u(1-\lambda_i), \quad \forall i \\
&z \;\; \ge x \cdot y^l + x_i \cdot (y - y^l) - M \cdot (1-\lambda_i), \quad \forall i \\
&z \;\; \ge x \cdot y^u + x_{i+1} \cdot (y - y^u) - M \cdot (1-\lambda_i), \quad \forall i \\
&z \;\; \le x \cdot y^u + x_i \cdot (y - y^u) + M \cdot (1-\lambda_i), \quad \forall i \\
&z \;\; \le x \cdot y^l + x_{i+1} \cdot (y - y^l) + M \cdot (1-\lambda_i), \quad \forall i \\
&x^l \le x \le x^u, \quad y^l \le y \le y^u
\end{aligned}
\tag{89}
$$

The convex hull formulation is defined by

$$
\begin{aligned}
\lambda_i &= \begin{cases} 1 & \text{if } x_i \le x \le x_{i+1} \\ 0 & \text{otherwise} \end{cases}, \forall i = 0, 1, \ldots, N-1 \\
&\textstyle\sum_{j=0}^{N-1} \lambda_i = 1 \\
x &= \textstyle\sum_{i=0}^{N-1} u_n \\
& x_i \cdot \lambda_i \le u_i \le x_{i+1} \cdot \lambda_i, \quad \forall i \\
y &= \textstyle\sum_{i=0}^{N-1} v_i \\
& y^l \cdot \lambda_i \le v_i \le y^u \cdot \lambda_i, \quad \forall i \\
z &\ge \textstyle\sum_{i=0}^{N-1} \left(u_i \cdot y^l + x_i \cdot v_i - x_i \cdot y^l \cdot \lambda_i \right) \\
z &\ge \textstyle\sum_{i=0}^{N-1} \left(u_i \cdot y^u + x_{i+1} \cdot v_i - x_{i+1} \cdot y^u \cdot \lambda_i \right) \\
z &\le \textstyle\sum_{i=0}^{N-1} \left(u_i \cdot y^l + x_{i+1} \cdot v_i - x_{i+1} \cdot y^l \cdot \lambda_i \right) \\
z &\le \textstyle\sum_{i=0}^{N-1} \left(u_i \cdot y^u + x_i \cdot v_i - x_i \cdot y^u \cdot \lambda_i \right) \\
& x^l \le x \le x^u, \quad y^l \le y \le y^u
\end{aligned}
\tag{90}
$$

A formulation representative of the category of incremental cost formulations is given by

$$
\begin{aligned}
\lambda_i &= \begin{cases} 1 & \text{if } x \ge x_{i+1} \\ 0 & \text{otherwise} \end{cases}, \quad 0 \le i \le N-2 \\
x &= x^l + \textstyle\sum_{i=0}^{N-1} (x_{i+1} - x_i) u_i \\
0 &\le u_i \le 1 \quad \forall i \\
z &= x^l \cdot y + y^l \cdot (x - x^l) + \textstyle\sum_{i=0}^{N-1} (x_{i+1} - x_i) \cdot w_i \\
w_i &\ge v_i, \quad \forall i < N-1 \\
w_0 &\ge (y^u - y^l) \cdot u_0 + y - y^l \\
w_i &\ge (y^u - y^l) \cdot (u_i - \lambda_{i-1}) + v_{i-1} \quad \forall i > 0 \\
w_0 &\le y - y^l \\
w_i &\le v_{i-1} \quad \forall i > 0 \\
w_{N-1} &\le (y^u - y^l) \cdot u_i \\
w_i &\le (y^u - y^l) \cdot (u_i - \lambda_i) + v_i, \quad \forall i < N-1 \\
& 0 \le w_{N-1} \le v_{N-2} \le w_{N-2} \le \cdots \le w_1 \le v_0 \le w_0 \le y - y^l
\end{aligned}
\tag{91}
$$

The authors in [86] also discuss and present numerical studies comparing formulations from these different categories using either identical segment lengths ($x_{i+1} - x_i = x_{j+1} - x_j, \forall i, j < N - 1$) or different segment lengths. The work in [45] is quite similar to the one mentioned in [86] but the authors include also numerical results comparing the piecewise-linear approximations when using the 'P'-formulation, which traces back to the work of Haverly ([48]), and the 'Q'-formulation (proposed by Ben-Tal et al. in [23]). In [65] different piecewise-linear relaxations are also studied for the bilinear pooling problem, here in the context of gas lifting operations.

3.1.2.2 Extensions for Cubic Problems

Here we present extensions of these works for problems with cubic constraints and discuss the different issues that can arise in this case. The work here is not as exhaustive as some of the works mentioned before, namely [86] and [45]. We restrict our selfs to models with one choice of binary variables (in this case the θ_i variable formulation), identical segment distance (that is, when partitioning one variable we always have $x_{i+1} - x_i = x_{j+1} - x_j, \forall i, j$), and the convex hull formulation (shown in Equation 90). For the bilinear terms in the model no modifications are introduced to the model shown in Equation 90.

To build piecewise linear approximations to trilinear terms we can use either the arithmetic intervals or recursive arithmetic intervals. We have implemented piecewise linear approximations using the recursive arithmetic intervals but we discuss what the possible advantages of using arithmetic intervals would be in this context. Although we are using the convex hull formulation to build the piecewise approximation for the bilinear terms, a straightforward extension of these approximations is not suitable here because of the large number of variables and constraints that would have to be introduced in the model. To see this we look at one of the constraints from the convex hull formulation in Equation 90:

$$z \geq \sum_{i=0}^{N-1} \left(u_i \cdot y^l + x_i \cdot v_i - x_i \cdot y^l \cdot \lambda_i \right) \tag{92}$$

This constraint corresponds to the constraint $z \geq y^l x + x^l y - x^l y^l$ and the variables λ_i, u_i, and v_i are added to the model for $i = 0, 1, \ldots, N - 1$. For each element of the partition of variable x we add to the model 3 sets of variables λ_i, u_i, and v_i. The variable λ_i is a

binary variable indicating if the value of x is in the current partition or not, u_i holds the value of x if x is in partition i and 0 otherwise, and v_i is a variable holding the value of y in the current domain, $[y^l, y^u]$, if x is in partition i and 0 otherwise. Let us now look at one of the constraints in the description of the recursive arithmetic intervals, namely the first constraint in Equation 80:

$$w_{xyz} \geq w_{xy}z^l + w_{xz}y^l + x^l w_{yz} - xy^l z^l - x^l yz^l - x^l y^l z + x^l y^l z^l \tag{93}$$

In order to extend the convex hull formulation to trilinear terms we would have to introduce the following variables into to the model: $\lambda_{xi}, u_{xi}, v_{yi}, v_{zi}, v_{xyi}, v_{xzi}$, and v_{yzi}. Some of these variables would also added to the model when doing the piecewise relaxation for the bilinear terms ($\lambda_{xi}, u_{xi}, v_{yi}, v_{zi}$), but the variables v_{xyi}, v_{xzi}, and v_{yzi} would have to be added for the relaxation of the trilinear terms. The corresponding constraint to be added to the model would be:

$$\begin{aligned} w_{xyz} \geq \sum_{i=0}^{N-1} \Big(& v_{xyi}z^l + v_{xzi}y^l + x_{i-1}v_{yzi} \\ & - u_{xi}y^l z^l - x_{i-1}z^l v_{yi} - x_{i-1}y^l v_{zi} + x_{i-1}y^l z^l \lambda_i) \end{aligned} \tag{94}$$

Assuming here N_t trilinear terms in the model, N_x variables involved in the trilinear terms and N partitions per partitioned variable, we have approximately $(N_t \cdot N_x^2 \cdot N)$ variables of the type v_{xyi} in the model. Adding these variables to the model we also have to add the constraints

$$\lambda_{xi} \cdot w_{yz}^l \leq v_{yzi} \leq \lambda_{xi} \cdot w_{yz}^u \tag{95}$$

The number of these constraints added is of the same order as the number of variable which in our case makes it impractical to use this formulation in the model. Instead we develop a formulation which is closer to a big$-M$ formulation but has some components similar to the convex hull formulation. The constraints added corresponding to the inequality in Equation 93 are

$$\begin{aligned} w_{xyz} \geq & x_{i-1}w_{yz} + y^l w_{xz} + z^l w_{xy} \\ & - y^l z^l u_{xi} - x_{i-1}z^l y - x_{i-1}y^l z + x_{i-1}y^l z^l - (1 - \lambda_{xi})M_{xyz} \quad \forall i \end{aligned} \tag{96}$$

These constraints have components from the convex hull formulation (namely the u_{xi} variables) but they are obviously a big-M type of formulation. The main difference from what would be the extension to trilinear terms from the formulation presented in Equation 89 would be that x is replaced by u_{xi} and the constraints

$$\begin{aligned} x &\geq x_i \cdot \lambda_i + x^l(1-\lambda_i), \quad \forall i \\ x &\leq x_{i+1} \cdot \lambda_i + x^u(1-\lambda_i), \quad \forall i \end{aligned} \tag{97}$$

would not be added to this model. It is easy to see that the constraints are valid for the trilinear term being approximated since when $\lambda_{xi} = 0$ the constraint is obviously redundant, as long as M_{xyz} is big enough, and when $\lambda_{xi} = 1$ the constraint reduces to the approximation that would be obtained if the variable x was defined in $[x_{i-1}, x_i]$. The constant M_{xyz} should be as small as possible, ideally we would have big-M value tailored for each x, y, z, i but in this case we claim that $M_{xyz} = 4$ is valid in general so we use that value. To see why it is valid we note that M_{xyz} must satisfy

$$M_{xyz} \geq x_i w_{yz} + y^l w_{xz} + z^l w_{xy} - x_{i-1} z^l y - x_{i-1} y^l z + x_{i-1} y^l z^l \tag{98}$$

for any values of $i, w_{yz}, w_{xy}, w_{xy}, y$, and z. In Equation 98 we have $x_i \leq x^u \leq 1, w_{yz} \leq y^u \cdot z^u \leq 1$ so $x_i w_{yz} + y^l w_{xz} + z^l w_{xy} + x_i y^l z^l \leq 4$. Since $0 \leq z^l, w_{xy}, x_i, y^l$, and z^l we have $-x_{i-1} z^l y - x_{i-1} y^l z \leq 0$ so the RHS in Equation 98 is bounded above by 4. In the context of this study we didn't explore other alternatives for the relaxation of the trilinear terms but if doing a more extensive study it would be interesting to try other alternatives. One that looks promising is the convex hull formulation using the arithmetic intervals, since with arithmetic intervals no additional variables are used, so we wouldn't have to define the extra variables v_{yzi} and additional constraints that come with the definition of these variables.

For a term of the form $x^2 y$ besides the different choices of which type of formulation and type of binary variable to use for the relaxation we also have to choose which of the variables x or y we choose to be the one we partition. As before we define a relaxation is an extension of the convex hull relaxation, and use the λ_i variables. If x is the variable which

we choose to partition we add the following constraints to the model:

$$\begin{aligned}
w_{xxy} &\geq \sum_{i=0}^{N-1} \left(2x_i y^l u_i + x_i^2 v_i - 2y^l x_i^2 \lambda_i\right) \\
w_{xxy} &\geq \sum_{i=0}^{N-1} \left(2x_{i+1} y^u u_i + x_{i+1}^2 v_i - 2y^u x_{i+1}^2 \lambda_i\right) \\
w_{xxy} &\geq \sum_{i=0}^{N-1} \left(x_i(y^l + y^u) u_i + x_i x_{i+1} v_i - \lambda_i x_i (x_i y^l + x_{i+1} y^u)\right) \\
w_{xxy} &\geq \sum_{i=0}^{N-1} \left(x_{i+1}(y^l + y^u) u_i + x_i x_{i+1} v_i - \lambda_i x_{i+1} (x_i y^l + x_{i+1} y^u)\right) \\
x &= \sum_{i=0}^{N-1} u_i \\
x_i \lambda_i \leq u_i \leq x_{i+1} \lambda_i, &\quad \forall i \\
y &= \sum_{i=0}^{N-1} v_i \\
y^l \lambda_i \leq v_i \leq y^u \lambda_i, &\quad \forall i \\
\sum_{i=0}^{N-1} \lambda_i &= 1 \\
\lambda_i \in \{0,1\}, &\quad \forall i
\end{aligned} \tag{99}$$

When choosing variable y to be the partitioned one, we can instead assume that we are building the relaxation for a term of the form xy^2 since in this case we don't have to change the notation for the points defining the partition x_i and the notation for the additional variables u_i and v_i.

$$\begin{aligned}
w_{xyy} &\geq \sum_{i=0}^{N-1} \left(2y^l x_i v_i + (y^l)^2 u_i + 2x_i (y^l)^2 \lambda_i\right) \\
w_{xyy} &\geq \sum_{i=0}^{N-1} \left(2x_{i+1} y^u v_i + (y^u)^2 u_i - 2x_{i+1} (y^u)^2 \lambda_i\right) \\
w_{xyy} &\geq \sum_{i=0}^{N-1} \left(y^l (x_i + x_{i+1}) v_i + y^l y^u u_i - y^l (y^l x_i + y^u x_{i+1}) \lambda_i\right) \\
y_{xyy} &\leq \sum_{i=0}^{N-1} \left(y^u (x_i + x_{i+1}) v_i + y^l y^u u_i - y^u (y^l x_i + y^u x_{i+1}) \lambda_i\right) \\
x &= \sum_{i=0}^{N-1} u_i \\
x_i \lambda_i \leq u_i \leq x_{i+1} \lambda_i, &\quad \forall i \\
y &= \sum_{i=0}^{N-1} v_i \\
y^l \lambda_i \leq v_i \leq y^u \lambda_i, &\quad \forall i \\
\sum_{i=0}^{N-1} \lambda_i &= 1 \\
\lambda_i \in \{0,1\}, &\quad \forall i
\end{aligned} \tag{100}$$

Note that the relaxations described in Equations 99 and 100 only address the computations of lower bounds. For the upper bound computation we start from the inequalities mentioned before

$$\begin{aligned} w_{xxy} &\leq (x^u)^2 y + (x - x^u)(x^l + x^u)y^l \\ w_{xxy} &\leq (x^l)^2 y + (x - x^l)(x^l + x^u)y^u \end{aligned} \tag{101}$$

and add the constraints corresponding to the relaxation of variable x. In a more comprehensive study we should also analyze the influence of discretizing variable y. The constraints added to Equations 99 and 100 for the discretization of variable x are:

$$\begin{aligned} w_{xxy} &\leq \sum_{i=0}^{N-1} \left(x_{i+1}^2 v_i + u_i(x_i + x_{i+1}) - x_{i+1}(x_i + x_{i+1})y^l \lambda_i \right) \\ w_{xxy} &\leq \sum_{i=0}^{N-1} \left(x_i^2 v_i + u_i(x_i + x_{i+1})y^u - x^l(x_i + x_{i+1})y^u \lambda_i \right) \end{aligned} \tag{102}$$

For the nonlinear terms x^2 the lower bounds are still given by adding subgradient inequalities at chosen points so the lower bound is in this case handled the same way as in Section 3.1.1. For the upper bounds for this nonlinear term we use the partitions defined before and the binary variables associated to these partitions and add secant inequalities corresponding to each of the partitions, as shown in Figure 8.

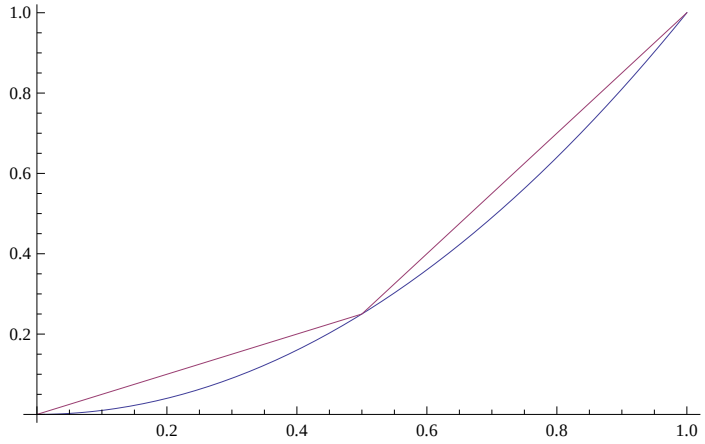

Figure 8: Upper Bound Piecewise Relaxation for x^2 with 2 points

Recalling that for x between a and b we have $x^2 \leq -ab + (a+b)x$ we can add constraints

to the model corresponding to a big$-M$ formulation:

$$\lambda_i = \begin{cases} 1 & \text{if } x_i \leq x \leq x_{i+1} \\ 0 & \text{otherwise} \end{cases} \qquad i = 0, \ldots, N-1 \tag{103}$$

$$z_{xx} \leq -x_i \cdot x_{i+1} + (x_i + x_{i+1})x + (1 - \lambda_i)M_i,$$

Each M_i has to be large enough to satisfy the inequality $x^2 \leq -x_i \cdot x_{i+1} + (x_i + x_{i+1})x + M_i$ which gives that M_i has to satisfy the inequalities

$$\begin{aligned} M_i \geq & (x^l)^2 + x_i x_{i+1} - (x_i + x_{i+1})x^l \\ M_i \geq & (x^u)^2 + x_i x_{i+1} - (x_i + x_{i+1})x^u \end{aligned} \tag{104}$$

we choose M_i to be the maximum of the two right hand sides in Equation 104. The other possibility here would be to use of an extension of the convex hull formulation, thus eliminating the need for the big$-M$ values. The constraints added to the model in this case are

$$\begin{aligned} z_{xx} \leq & \sum_{i=0}^{N-1} (-x_i \cdot x_{i+1}\lambda_i + (x_i + x_{i+1})u_i) \\ x_i \cdot \lambda_i \leq u_i \leq & x_{i+1} \cdot \lambda_i \quad i = 0, \ldots, N-1 \\ x = & \sum_{i=0}^{N-1} u_i \end{aligned} \tag{105}$$

3.2 Results of Upper Bounds Techniques

Here we describe the results obtained for computing upper bounds for problem **MaxAvg**. After a feasible solution is obtained we wish to estimate the quality of this solution. For this we use models described in Section 3.1. The non-linear constraints are replaced the linear approximations and the resulting model is solved with **CPLEX**, version 12.2. We compute the resulting gap between the feasible solution resulting from the heuristic and the upper bound by

$$gap = \frac{u - h}{h} \tag{106}$$

where in Equation 106 h is the objective function value of the solution obtained with the heuristic procedure and u the objective function value of the problem providing an upper bound. When using the relaxation described in Section 3.1.1 the resulting problem solves

very quickly but the bound obtained is very big (around 50%), so this bound is not of any usefulness in evaluating the quality of the solutions we obtain with the primal heuristic.

The relaxations from Section 3.1.2 give much tighter bounds at the expense of a much higher computational time. There are essentially 3 groups of constraints for which we use the techniques described in that section: the pooling constraints, the stability constraints, and the constraints for cleaning model. Since each of these groups of constraints are defined by 3 different groups of variables we have to use 3 groups of extra variables to model the piecewise linear formulations of the constraints. For each variable partitioned in the pooling constraints we add to the model 5 additional binary variables. Recalling that the pooling constraints are defined by

$$q_{il} \cdot y_{lj} = v_{ilj} \quad \forall i \in I^{RM}, l \in I^{P}, j \in FP \tag{107}$$

we use the fact the variables y_{lj} have a smaller range and there are substantially less of them when compared with the variables q_{il} to choose the variables y_{lj} the ones to be partitioned. Each of the variables in the stability model is approximated by an extra 5 binary variables while each partitioned variable in the cleaning model we introduce 7 extra binary variables in the model. There are obviously extra variables introduced in the model as mentioned in Section 3.1.2. Applying the techniques described in the mentioned section to problem **MaxAvg** reduces the gap between the upper bound and the feasible solution to about 9.80%, after an hour of computation time. This is much better than the 50% we reported without using the partition scheme, but it is still high. We note however that after one hour the **MIP** was not solved to optimality, the gap we report is the gap computed with the best upper bound known to **CPLEX** when the maximum computation time was reached. The best feasible solution for the approximation problem was at that point 1.32%. So our true gap lies in the interval $[1.32, 9.80]$. In order to get a better idea how the approximation scheme would perform we applied the approximation scheme to subproblems of the original problem. The subproblems we consider have 1 pool and 2 final products. In Table 14 we show the lower and upper bounds for the gap given by this type of relaxation (the lower bound on the gap is given by using the best feasible solution and the lower upper

bound known to **CPLEX** at the time limit). The reported value in the tables is computed according Equation 106 multiplied by 100. When averaged over all the combinations of final

Table 14: Sample Results for Convex Hull Approximation

	Lower Bound	Upper Bound
(1, 2)	1.70%	4.14%
(1, 3)	1.60%	3.74%
(1, 4)	1.69%	4.21%
(1, 5)	1.34%	2.98%
(1, 6)	3.62%	4.53%
(1, 7)	3.82%	6.13%
(1, 8)	5.29%	5.85%
(1, 9)	1.80%	3.02%
(1, 10)	1.45%	1.96%

products the interval for lower and upper bound that we can achieve by using this method is $[2.12, 3.24]\%$.

We now present, in Table 15, some results pertaining to the variable which should be chosen for partitioning when considering terms of the form x^2y. The average gap over all the problems is 3.50% for the version where we partition the variable x and 5.68% for the version where we partition the variable y.

Table 15: Gap when Partitioning x or y in x^2y

	% Gap Part. x	% Gap Part. y
(1,2)	4.037	7.905
(1,3)	4.349	7.66
(1,4)	4.567	6.962
(1,5)	3.592	5.565
(1,6)	3.67	6.525
(1,7)	7.334	10.434
(1,8)	6.359	8.96
(1,9)	3.846	6.093
(1,10)	3.68	6.095

3.3 Summary

In this chapter we looked at the problem of computing upper bounds for the industrial cubic problem defined in Chapter 2. We described the general form of the problems used to obtain

the bounds and which specific functions are used in the approximation of the non-linear terms in the model. For some of the non-linear terms we provide the error between the approximation and the term with respect to the lower and upper bounds of the variables. We recall the standard approximations for trilinear terms and discuss which of these are suitable to use in our case. In the last part of the chapter we extended the work in [45, 86] for cubic polynomial problems.

CHAPTER IV

CONVEX RELAXATIONS FOR CUBIC POLYNOMIAL PROBLEMS

As seen in Chapter 2 problems with cubic constraints provide the ability to model complicated industrial processes. Solving these cubic problems is usually a very challenging task due to the hardness of this class of problems (it is a well know fact that quadratic problems are NP-hard, even if just one negative eigenvalue is present (see [68, 75]), so trivially cubic problems are NP-Hard as well. It is also well know that it is possible to solve higher order problems involving polynomial functions by introducing new variables and equalities relating the new variables and the non-linear terms in the original model. As an example suppose we have a trilinear term xyz in the model. Introduce two new variables w_0 and w_1, replace all occurrences of xyz by w_0 and add to the model the two equalities: $w_0 = w_1 z$, $w_1 = xy$. The reformulated model is easily seen to be equivalent to the initial one. We study here properties of relaxations for the non-linear terms in a cubic problem, specially for the terms of the form $x^2 y$, where x and y are non-negative continuous bounded variables. In this work we look at relaxations obtained via the mentioned reformulation and relaxations tailored for specifically for cubic problems. We have a section devoted for the $[0, 1]$ case since some of the results can be simplified or given more accurately, and a section for the general non-negative case.

4.1 Definitions

We present here definitions for concepts that we use throughout the text.

Consider a function $f : D \subseteq \mathbb{R}^n \to \mathbb{R} \cup \{-\infty, \infty\}$.

Definition 1 (Epigraph). *The epigraph of f is the set*

$$\{(x, \mu) \mid x \in D,\ \mu \in \mathbb{R},\ \mu \geq f(x)\} \tag{108}$$

Definition 2 (Underestimating Funtion (or Underestimator)). *$\underline{f}$ is an underestimating function for f if $\underline{f}(x) \leq f(x), \forall x \in D$.*

Definition 3 (Convex Envelope / Concave Envelope). *F is the convex envelope (or convex hull) of f if F is a convex underestimator of f and if $\tilde{F}$ is another convex underestimator of f we have $\tilde{F}(x) \leq F(x)$. An alternative characterization of F is given by defining*

$$F(x) = \inf\Big\{\lambda_1 f(x_1) + \lambda_2 f(x_2) + \cdots + \lambda_n f(x_n) \mid x_1, x_2, \ldots, x_n \in D, \\ x = \sum_{i=1}^{n} \lambda_i x_i, \sum_{i=1}^{n} \lambda_i = 1, \lambda_i \geq 0, \forall i\Big\} \quad (109)$$

Concave envelope is defined in a similar way by using the definition of concave function.

Definition 4 (Gap of underestimator). *The Gap of an underestimator function of f, $\underline{f}$ is given by*

$$Gap_D(f, \underline{f}) = \max_{x \in D}\big(f(x) - \underline{f}(x)\big) \quad (110)$$

4.2 Previous Work

McCormick in [62] gives the convex envelope for a bilinear term xy with x and y defined in a rectangle, $(x, y) \in [x^l, x^u] \times [y^l, y^u]$ with x^l, x^u, y^l,and y^u finite. The author also discusses how to obtain convex underestimating problems for any problem which is a factorable problem. Al-Khayyal and Falk [17] develop a procedure, which converges in an infinite number of steps, for constrained bilinear problems using lower bounds derived from the convex envelopes of the bilinear terms xy over rectangular regions. In this work they show that the convex hull of $f(x, y) = xy$ in a rectangular region are the McCormick envelopes. Sherali and Tuncbilek [76] compare strategies for constructing linear programming relaxations for polynomial programming problems using a Reformulation-Linearization Technique (RTL). The basic idea of the Reformulation-Linearization technique is that redundant constraints are added to the problem (for example given a constraint $g(x) \leq 0$, and a valid inequality $x_1 - x_1^l \geq 0$ then the valid (but redundant) inequality $(x_1 - x_1^l)g(x) \leq 0$ would be added to the problem and the non-linear term are subsequently linearized). Li et al. [58] and later Lu et al. [59] give convex underestimators for posynomial functions of positive variables (a function f is posynomial if $f(x) = dx_1^{\alpha^1} dx_2^{\alpha^2} \cdots dx_n^{\alpha^n}, 0 < x_i^l \leq x_i \leq x_i^u, \alpha_i \in \mathbb{R}$ for $i = 1, 2, \ldots, n$, and $d > 0$). Jach, Michaels, and Weismantel [53] show that evaluating the

value of the convex envelope of an $(n-1)-$convex indefinite function on a box is computationally tractable as long as the number of variables in the function definition is small. An actual analytical expression of the convex envelope is only possible for specific families of functions. An $(n-1)$-convex indefinite function is a function which is convex when one of its variables is fixed at a specific value and for each point in the domain the Hessian of the function is indefinite (it has both negative and positive eigenvalues). Maranas and Floudas [60], Adjiman et al. [16], Adjiman et al. [15], and Floudas [36] give theoretical and computational properties of the αBB method. In this method convex underestimators for twice continuously differentiable functions f are obtained by adding a quadratic term to the function so that the resulting function is convex. The resulting function is

$$\tilde{f}(x) = f(x) + \alpha(x^L - x)^T(x^U - x) \tag{111}$$

where α is a positive parameter that makes the overall function convex. In order to have the underestimator as tight as possible the parameter α is estimated by using interval arithmetic techniques on the Hessian matrix or the characteristic polynomial of the function being investigated. Gounaris and Floudas [43], [44] presented a piecewise application of the $\alpha-$BB method that produces tighter convex underestimator than the original variants. The first paper deals with univariate functions and the second one with multivariate functions. Convex relaxations for quadrilinear terms are analyzed by Cafieri, Lee, and Liberti [30]. Nick Sahinidis and Tawarmalani in [81] define the notion of convex extensions of lower semi-continuous functions (a lower semi-continuous function or $l.s.c.$ function is a function $f : \mathbb{R}^n \to \mathbb{R}$ whose epigraph is a closed set in $\mathbb{R}^{n+1}$). They give conditions for the existence of these convex extensions and characterize the tightest one when multiple convex extensions exist. Using the theory of convex extensions they analyze the gap of various underestimator for the function x/y over a rectangle and show that the obtained relaxation is tighter than the classical ones obtained by outer-linearization of bilinear terms. A continuation of this work is presented in [80] where the convex and concave envelopes of x/y are given over a hypercube. They also show the convex envelope for a class of functions of the type $f(x, y)$ where f is concave in x and convex in y. Bao, Sahinidis, and Tawarmalani [21]

present multiterm polyhedral relaxations for nonconvex, quadratically constrained quadratic problems (the relaxations are derived for a sum of bilinear terms instead of the traditional relaxations that look at a term at a time). Kilinc-Karzan [55] presents relaxations for a sum of bilinear terms $\sum_i u_i x_i$ where the variables u and x are disjoint.

4.3 The unit interval case

Denote $S = [x^l, x^u] \times [y^l, y^u] = [0,1] \times [0,1]$, $\phi(x,y) = x^2 y$ and $\phi_2(x,y) = \max\{x^2 + y - 1, 0\}$.

Result 1. *ϕ_2 is a convex underestimator of ϕ in S.*

Proof. We note that it is immediate that ϕ_2 is convex since it is the maximum of two convex functions. To show that it is an underestimator of ϕ in S we have

$$0 \leq (x - x^l)(x^u - x)(y^u - y) = (1 - x)x(1 - y) = x^2 y - x^2 - yx + x \tag{112}$$

and from here $x^2 y \geq x^2 - x + xy$. Since xy is not convex we use the McCormick convex underestimators defined in [17],

$$xy \geq \begin{cases} xy^l + x^l y - x^l y^l \\ xy^u + x^u y - x^u y^u \end{cases} \tag{113}$$

which for the current bounds x^l and y^l amounts to $xy \geq \max\{0, x + y - 1\}$. By replacing xy with this expression we get $x^2 y \geq x^2 - x + \max\{0, x + y - 1\} = \max\left\{x^2 - x, x^2 + y - 1\right\}$. In the same way

$$0 \leq (x^u - x)^2 (y - y^l) = (1 - x)^2 y = x^2 y - 2xy + y \tag{114}$$

so $x^2 y \geq 2xy - y \geq 2\max\{0, x + y - 1\} - y = \max\{-y, 2x + y - 2\}$. We also have obviously that $x^2 y \geq 0$ in S. At this point we have that $\max\left\{0, -y, x^2 - x, 2x + y - 2, x^2 + y - 1\right\}$ is a valid underestimator for $x^2 y$ in S, but since $x^2 - x$ and $-y$ are non positive on S they can be ignored. From $(x^2 + y - 1) - (2x + y - 2) = x^2 - 2x + 1 = (x - 1)^2 \geq 0$ we conclude that $2x + y - 2$ is also redundant. □

Theorem 1. *Let $\phi(x,y) = x^2 y$. The convex envelope $\phi_{conv}(x,y)$ of ϕ in S is given by*

$$\phi_{conv}(x,y) = \begin{cases} 0 & \text{if } x + y \leq 1 \\ \frac{(x-1)^2}{y} + 2(x-1) + y & \text{if } x + y > 1 \end{cases} \tag{115}$$

Proof. Following the work in [80] we note the ϕ is convex in x for fixed y in $[0,1]$ and concave in y for fixed x. By using disjunctive programming techniques (see [72]) we have that the convex hull of the epigraph is ϕ in the domain is defined as

$$\begin{cases} z & \geq & (1-\lambda)y^l x_a^2 + \lambda y^u x_b^2 \\ x & = & (1-\lambda)x_a + \lambda x_b \\ y & = & (1-\lambda)x^l + \lambda y^u \\ 0 \leq x_a, x_b \leq 1 \\ 0 \leq \lambda \leq 1 \end{cases} \tag{116}$$

Given x and y we want to find smallest z in the above set corresponding to this pair (x, y). Since $y^l = 0$ and $y^u = 1$ we want to solve the problem:

$$\begin{array}{rll} \min_{x_a, x_b} & \lambda x_b^2 & \\ \text{subject to:} & x = & (1-\lambda)x_a + \lambda x_b \\ & y = & \lambda \\ & 0 \leq x_a, x_b \leq 1 & \\ & 0 \leq \lambda \leq 1 & \end{array} \tag{117}$$

The objective function value of problem in Equation 117 is 0 whenever x_b is 0 and this amounts to having $x = (1-\lambda)x_a$ or equivalently $x = (1-y)x_a$ from $y = \lambda$. From $x_a \leq 1$ we get that this is valid for $y + x \leq 1$. For $y + x > 1$ we have x_b is given by $(x + y - 1)/y$. Replacing this in the objective function we get that the value of the program is given by $y((x + y - 1)/y)^2$ which simplifies to $\frac{(x-1)^2}{y} + 2(x-1) + y$. □

We want to measure what is the error in underestimating ϕ by ϕ_{conv} and by ϕ_2. We consider maximum gap and average gap.

Result 2. $\max_{(x,y)\in S}(\phi(x,y) - \phi_{conv}(x,y)) = 4/27$.

Proof. If $x + y \leq 1$ then $\phi(x,y) - \phi_{conv}(x,y)) = x^2 y$. Note that in the domain $x^2 y$ is increasing in both x and y so $\max_{\{(x,y)\in S | x+y\leq 1\}}(\phi(x,y) - \phi_{conv}(x,y))$ is in a boundary point. In $x = 0$ or $y = 0$ the value of the function is 0. In the line segment $\{(x,y) \in S \mid x + y = 1\}$ we have $x^2 y = x^2(1-x)$ and the maximum value in this line segment is $4/27$

at $(x, y) = (2/3, 1/3)$ (this is easily seen by differentiating $x^2 - x^3$ and finding the zero of the derivative). We consider now $\max_{\{(x,y)\in S | x+y>1\}}(\phi(x, y) - \phi_{conv}(x, y))$.In this case we have $\phi(x, y) - \phi_{conv}(x, y)) = x^2y - (x - 1)^2/y - 2(x - 1) - y$. With a similar reasoning to the previous part we note that $\frac{\partial}{\partial x}(\phi(x, y) - \phi_{conv}(x, y)) = 2(xy - x/y - 1/y - 1) = 2(x(y - 1/y) - 1/y - 1) = (2/y)(x(y^2 - 1) - (y + 1)) = (2/y)(y + 1)(x(y - 1) - 1)$. The previous expression is 0 only if $y = -1$ or $x = 1/(y - 1)$. For here we conclude that $\frac{\partial}{\partial x}(\phi(x, y) - \phi_{conv}(x, y)) < 0$ for $\{(x, y) \in S \mid x + y > 1\}$ and the maximum value must again be obtained in a boundary point. Doing the similar computations we find again the optimal value to be $4/27$ at the point $(x^*, y^*) = (2/3, 1/3)$. □

Result 3. $\max_{(x,y)\in S}(\phi(x, y) - \phi_2(x, y)) = 1/4$.

Proof. We want now to compute the maximum gap between $\phi(x, y)$ and $\phi_2(x, y)$ in S. If $(x, y) \in S$ are such that $x^2 + y - 1 \leq 0$ then $\phi(x, y) - \phi_2(x, y) = x^2y$ increasing both in x and y. The maximum value in $\{(x, y) \in S \mid x^2 + y - 1 \leq 0\}$ is then attained along the curve segment $\{(x, y) \in S \mid x^2 = 1 - y\}$ from which we conclude that the optimal value is $1/4$ attained at $(\tilde{x}, \tilde{y}) = (1/2, 1/2)$ Considering now the case $(x, y) \in S$ and $x^2 + y - 1 > 0$ we have $\phi(x, y) - \phi_2(x, y) = x^2y - x^2 - y + 1$. Differentiating with respect to x we find that the partial derivative is negative in the interior of this region so the maximum value is attained in the boundary and we get the same value for the maximum gap, $1/4$ at $(\tilde{x}, \tilde{y})$. □

Result 4. $\iint_S(\phi(x, y) - \phi_{conv}(x, y))dS = 1/18$.

Result 5. $\iint_S(\phi(x, y) - \phi_2(x, y))dS = 1/15$.

The above integrals were computed in **MATHEMATICA** [87].

Result 6. $\max_{(x,y)\in S}(\phi_{conv}(x, y) - \phi_2(x, y)) = 4/27$.

Proof. The computation for this value is similar to previous computations. In this case we divide S in three regions:

$$\begin{aligned} S^1 = &\quad S \bigcap \{(x, y) \mid x + y \leq 1\} \\ S^2 = &\quad S \bigcap \{(x, y) \mid x + y > 1 \text{ and } x^2 + y - 1 < 0\} \\ S^3 = &\quad S \bigcap \{(x, y) \mid x^2 + y - 1 >= 0\} \end{aligned} \tag{118}$$

On S^1 both functions are 0. On S^2, ϕ_2 is 0 so the maximum in S^2 is given by the previous result 4/27. On S^3 we analyze the expression $(x-1)^2/y + 2(x-1) + y - (x^2 + y - 1)$, conclude that it is decreasing in x, compute the maximum value along the curve $S \bigcap \left\{(x,y) \mid x^2 + y - 1 = 0\right\}$ and conclude that the optimum value (at $(x,y) = \left(\left(-1+\sqrt{5}\right)/2, \left(-1+\sqrt{5}\right)/2\right)$) is smaller than 4/27 so the result follows. □

Result 7. $\iint_S (\phi_{conv}(x,y) - \phi_2(x,y))dS = 1/90$

The above result was again computed in **MATHEMATICA** [87].

4.4 General (positive) Bounds

We consider now the more general case where instead of $(x,y) \in [0,1] \times [0,1]$ we have $(x,y) \in [x^l, x^u] \times [y^l, y^u]$, with $x^l, y^l \geq 0$. We show first how the underestimators are obtained. By using an analogous procedure to the one used for deriving the McCormick envelopes for bilinear terms we start from

$$0 \leq (x - x^l)(x - x^l)(y - y^l) \tag{119}$$

expanding the RHS and rearranging the terms we get

$$x^2 y \geq y^l x^2 + 2x^l xy - 2x^l y^l x - (x^l)^2 y + (x^l)^2 y^l \tag{120}$$

There are two nonlinear terms in the right hand side of inequality in Equation 120. The term $y^l x^2$ is convex since $y^l \geq 0$, but the term $2x^l xy$ is nonconvex for all $x^l \neq 0$. If we use the McCormick envelopes $xy \geq \max\{y^l x + x^l y - x^l y^l, y^u x + x^u y - x^u y^u\}$ we arrive at

$$x^2 y \geq \max \begin{cases} y^l x^2 + (x^l)^2 y - (x^l)^2 y^l \\ y^l x^2 + 2x^l (y^u - y^l)x + x^l (2x^u - x^l)y + \left((x_l)^2 y^l - 2x^l x^u y^u\right) \end{cases} \tag{121}$$

We denote by F_{11} and F_{12} the first and second functions in the RHS of Equation 121, that is F_{11} is the function we obtain by replacing xy by $y^l x + x^l y - x^l y^l$ in the RHS of Equation 120 and F_{12} is the function we obtain by replacing xy by $y^u x + x^u y - x^u y^u$ in the RHS of Equation 120. Proceeding the same way for the inequality

$$0 \leq (x^u - x)(x - x^l)(y^u - y) \tag{122}$$

we arrive at functions F_{21} and F_{22}, while starting from

$$0 \leq (x^u - x)(x^u - x)(y - y^l) \tag{123}$$

we define the functions F_{31} and F_{32}.

For reference :

$$\begin{aligned}
F_{11}(x,y;x^l,x^u,y^l,y^u) &= y^l x^2 + (x^l)^2 y - (x^l)^2 y^l \\
F_{12}(x,y;x^l,x^u,y^l,y^u) &= y^l x^2 + 2x^l(y^u - y^l)x + x^l(2x^u - x^l)y \\
&\qquad + x^l(x^l y^l - 2x^u y^u) \\
F_{21}(x,y;x^l,x^u,y^l,y^u) &= y^u x^2 + ((x^l + x^u)(y^l - y^u))x \\
&\qquad + (x^l)^2 y + x^l(x^u y^u - (x^l + x^u)y^l) \\
F_{22}(x,y;x^l,x^u,y^l,y^u) &= y^u x^2 + (x^u)^2 y - (x^u)^2 y^u \\
F_{31}(x,y;x^l,x^u,y^l,y^u) &= y^l x^2 + x^u(2x^l - x^u)y + x^u y^l(x^u - 2x^l) \\
F_{32}(x,y;x^l,x^u,y^l,y^u) &= y^l x^2 + 2x^u(y^u - y^l)x + (x^u)^2 y + (x^u)^2(y^l - 2y^u)
\end{aligned} \tag{124}$$

We can now naturally define an underestimator F_{MC} for $f(x,y) = x^2 y$ by

$$F_{MC}(x,y) = \max\{F_{11}(x,y), F_{12}(x,y), F_{21}(x,y), F_{22}(x,y), F_{31}(x,y), F_{32}(x,y)\} \tag{125}$$

Theorem 2. *The function F_{MC} defined in Equation 125 is a convex underestimator for $f(x) = x^2 y$ for $(x,y) \in [x^l, x^u] \times [y^l, y^u]$, with $x^l \geq 0$ and $y^l \geq 0$.*

Proof. We note that each function in the max definition of F_{MC} is an underestimator of $f(x)$ because of the way these inequalities are derived, so the maximum of these functions is also an underestimator of $f(x)$. Also each of the functions in the definition is convex and so F_{MC} as the maximum of convex functions. □

Theorem 3 (Redundancy in the definition of F_{MC}). *For $x, y \in [x^l, x^u] \times [y^l, y^u]$ with $x^l \geq 0$ and $y^l \geq 0$ we have*

$$F_{MC}(x,y) = \max\{F_{11}(x,y), F_{12}(x,y), F_{22}(x,y)\} \tag{126}$$

If $x^l = 0$ then $F_{11} = F_{12}$ and so in this case one this functions is also redundant.

Proof. To show that F_{21} is redundant we show that that

$$F_{21}(x, y; x^l, x^u, y^l, y^u) \leq F_{11}(x, y; x^l, x^u, y^l, y^u) \tag{127}$$

We have

$$\begin{aligned} F_{11}(x, y; x^l, x^u, y^l, y^u) - F_{21}(x, y; x^l, x^u, y^l, y^u) = \\ &= (y^l - y^u)x^2 + (x^l(y^u - y^l) + x^u(y^u - y^l))x - x^l x^u(y^u - y^l) \\ &= (y^u - y^l)(-x^2 + x(x^l + x^u) - x^l x^u) \\ &= (y^u - y^l)(x^l(x - x^u) + xx^u - x^2 \\ &= (y^u - y^l)(x^l(x - x^u) + x(x^u - x) \\ &= (y^u - y^l)(x^u - x)(x - x^l) \end{aligned} \tag{128}$$

By looking at the last expression it is immediate than it is nonnegative since $y^u \geq y^l$ and $x \in [x^l, x^u]$ which verifies the inequality in Equation 127. We now show that F_{31} is redundant by showing

$$F_{31}(x, y; x^l, x^u, y^l, y^u) \leq F_{11}(x, y; x^l, x^u, y^l, y^u) \tag{129}$$

We have

$$\begin{aligned} F_{11}(x, y; x^l, x^u, y^l, y^u) - F_{31}(x, y; x^l, x^u, y^l, y^u) = \\ &= (x^u - x^l)^2 y - (x^u - x^l)^2 y^l \\ &= (x^u - x^l)^2 (y - y^l) \end{aligned} \tag{130}$$

It is also clear that this expression in nonnegative for $y \geq y^l$ which verifies the inequality in Equation 129. To show that F_{32} is redundant we proceed in the same way now showing that

$$F_{32}(x, y; x^l, x^u, y^l, y^u) \leq F_{22}(x, y; x^l, x^u, y^l, y^u) \tag{131}$$

We have

$$\begin{aligned} F_{22}(x, y; x^l, x^u, y^l, y^u) - F_{32}(x, y; x^l, x^u, y^l, y^u) = \\ &= (y^u - y^l)x^2 + 2x^u(y^l - y^u)x + (x^u)^2(y^u - y^l) \\ &= (y^u - y^l)(x^2 - 2x^u x + (x^u)^2) \\ &= (y^u - y^l)(x - x^u)^2 \end{aligned} \tag{132}$$

which is clearly nonnegative and thus verifies the inequality in Equation 131. We now show that F_{11}, F_{12} and F_{22} are not redundant in the definition of F_{MC}. To show that F_{11} is not redundant we exhibit points x, y and parameters x^l, x^u, y^l, y^u such that

$$\begin{aligned} F_{12}(x,y;x^l,x^u,y^l,y^u) &< F_{11}(x,y;x^l,x^u,y^l,y^u) \\ F_{21}(x,y;x^l,x^u,y^l,y^u) &< F_{11}(x,y;x^l,x^u,y^l,y^u) \end{aligned} \tag{133}$$

Taking $x = 0.1$, $y = 0.1$, $x^l = 0.1$, $x^u = 0.5$, $y^l = 0.1$ and $y^u = 0.5$ we have

$$\begin{aligned} F_{11}(0.1,0.1;0.1,0.5,0.1,0.5) - F_{12}(0.1,0.1;0.1,0.5,0.1,0.5) &= 0.032 \\ F_{11}(0.1,0.1;0.1,0.5,0.1,0.5) - F_{22}(0.1,0.1;0.1,0.5,0.1,0.5) &= 0.096 \end{aligned} \tag{134}$$

Which shows that we cannot eliminate F_{11} from the definition of F_{MC}.
To show that F_{12} is necessary in the definition of F_{MC} we proceed in the same way exhibiting a point where F_{21} dominates F_{11} and F_{22}. We take now $x = 0.3$, $y = 0.35$, $x^l = 0.1$, $x^u = 0.5$, $y^l = 0.1$, $y^u = 0.5$ which gives

$$\begin{aligned} F_{12}(0.3,0.35;0.1,0.5,0.1,0.5) - F_{11}(0.3,0.35;0.1,0.5,0.1,0.5) &= 0.004 \\ F_{12}(0.3,0.35;0.1,0.5,0.1,0.5) - F_{22}(0.3,0.35;0.1,0.5,0.1,0.5) &= 0.008 \end{aligned} \tag{135}$$

In order to show that F_{22} is not redundant we consider the points $x = 0.5$, $y = 0.5$, $x^l = 0.1$, $x^u = 0.5$, $y^l = 0.1$, $y^u = 0.5$ which gives

$$\begin{aligned} F_{22}(0.5,0.5;0.1,0.5,0.1,0.5) - F_{11}(0.5,0.5;0.1,0.5,0.1,0.5) &= 0.096 \\ F_{22}(0.5,0.5;0.1,0.5,0.1,0.5) - F_{12}(0.5,0.5;0.1,0.5,0.1,0.5) &= 0.064 \end{aligned} \tag{136}$$

Finally when $x^l = 0, F_{11} = y^l x^2 = F_{12}$ so one of these functions is redundant.

□

Definition 5 (Consistent Estimator). *We say an underestimator φ for f is consistent on a set D if*

$$\sup_{x \in D} |f(x) - \varphi(x)| \to 0 \text{ when } diam(D) \to 0 \tag{137}$$

where $diam(D) = \sup\{\|x - y\| \mid x, y \in D\}$.

Theorem 4. *The convex estimator F_{MC} defined in Equation 126 is consistent for $f(x,y) = x^2 y$ in $D = [x^l, x^u] \times [y^l, y^u]$.*

Proof. We have $x^2y - F_{11}(x,y;x^l,x^u,y^l,y^u) = x^2y - y^lx^2 - (x^l)^2y + (x^l)^2y^l \to (x^l)^2y^l - (x^l)^2y^l - (x^l)^2y^l + (x^l)^2y^l = 0$ since if $diam([x^l,x^u]\times[y^l,y^u]) \to 0$ we must have $x^u - x^l \to 0$ and $y^u - y^l \to 0$ which immediately gives $x^2y \to (x^l)^2y^l$, $y^lx^2 \to (x^l)^2y^l$ and $(x^l)^2y \to (x^l)^2y^l$. We omit the analysis for the functions F_{12} and F_{22} since it is similar. The result follows from noting that $\left|F_{MC}(x,y) - x^2y\right| \leq \max\{\left|F_{11}(x,y) - x^2\right|, \left|F_{12}(x,y) - x^2\right|, \left|F_{22}(x,y) - x^2\right|\}$ and each of the terms in the parenthesis goes to zero as $diam(D) \to 0$. □

Theorem 5 (Gap of F_{MC}). *The gap for the underestimator F_{MC} is the following:*

$$\begin{cases} (x - x^l)(x + x^l)(y - y^l), & (x,y) \in \mathcal{R}_1 \\ x^2y - y^lx^2 - 2x^l(y^u - y^l)x - x^l(2x^u - x^l)y + x^l(2x^uy^u - x^ly^l), & (x,y) \in \mathcal{R}_2 \\ ((x^u)^2 - x^2)(y^u - y), & (x,y) \in \mathcal{R}_3 \end{cases} \tag{138}$$

where

$$\mathcal{R}_1 = \left\{(x,y) : y \leq y^u + \frac{y^l - y^u}{x^u - x^l}(x - x^l)\right\} \tag{139}$$

$$\mathcal{R}_2 = \left\{(x,y) : y^u + \frac{y^l - y^u}{x^u - x^l}(x - x^l) \leq y \leq \frac{-(y^u - y^l)x^2 + (x^u)^2y^u - (x^l)^2y^l}{(x^u)^2 - (x^l)^2}\right\} \tag{140}$$

$$\mathcal{R}_3 = \left\{(x,y) : y \geq \frac{-(y^u - y^l)x^2 + (x^u)^2y^u - (x^l)^2y^l}{(x^u)^2 - (x^l)^2}\right\} \tag{141}$$

Furthermore the gap of F_{MC} does not exceed

$$\max\left\{\frac{2(3(x^l)^2(-3x^u + \hat{x}) + (x^u)^2(x^u + \hat{x}))(y^u - y^l)}{27(x^u - x^l)},\ \frac{(x^u - x^l)(x^u - x^l)(y^u - y^l)}{2}\right\} \tag{142}$$

where $\hat{x} = \sqrt{3(x^l)^2 + (x^u)^2}$.

Proof. We start by showing pictorially how the proof is going to be. First we show that for any domain of the form $[x^l,x^u]\times[y^l,y^u]$ our underestimators are dominant in a well defined region. Next we compute the approximation error that is made if the considered point is in one of the parts of the domain and finally we give the maximum error that can be computed for each of the domain parts. We can see how the domain is divided in Figure 9.

We claim that F_{11} that it is the dominant piece if

$$y - y^u \leq \frac{y^l - y^u}{x^u - x^l}(x - x^l) \tag{143}$$

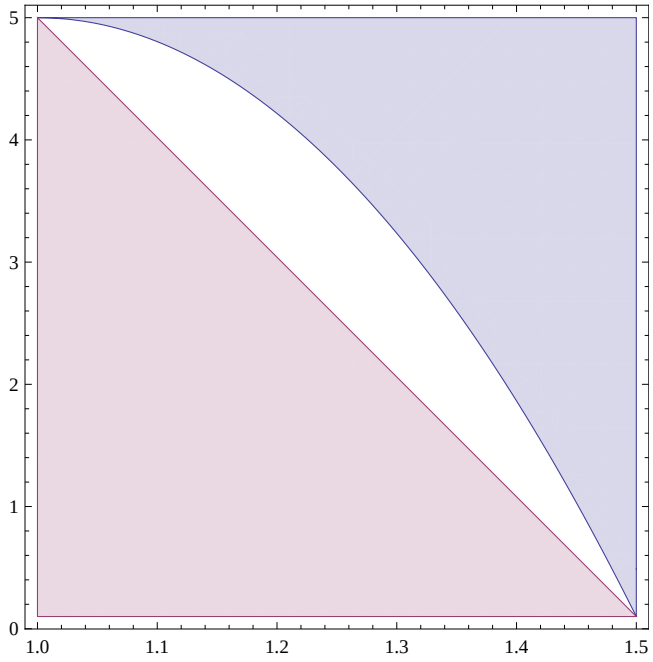

Figure 9: Regions where the different approximations are valid

To see this we start by noting that

$$F_{11}(x,y;x^l,x^u,y^l,y^u) - F_{12}(x,y;x^l,x^u,y^l,y^u) \\ = 2x^l(-(y^l-y^l)x-(x^u-x^l)y+(x^uy^u-x^ly^l)) \quad (144)$$

From here we see that the difference $F_{11}-F_{12}$ decreases when either x or y increase. When we evaluate the difference computed at the points of the line $\{(x,y) : y = y^u + \frac{y^l-y^u}{x^u-x^l}(x-x^l)\}$ we get $F_{11}(x,y;x^l,x^u,y^l,y^u) - F_{12}(x,y;x^l,x^u,y^l,y^u) = 0$.

Also

$$F_{11}(x,y;x^l,x^u,y^l,y^u) - F_{22}(x,y;x^l,x^u,y^l,y^u) \\ = -(y^u-y^l)x^2-(x^u+x^l)(x^u-x^l)y+((x^u)^2y^u-(x^l)^2y^l) \quad (145)$$

and here also since the RHS of Equation 145 decreases with both x and y and again computing the value of the difference at the line $\{(x,y) : y = y^u + \frac{y^l-y^u}{x^u-x^l}(x-x^l)\}$ we get $F_{11}(x,y;x^l,x^u,y^l,y^u) - F_{22}(x,y;x^l,x^u,y^l,y^u) = (x-x^l)(x^u-x)(y^u-y^l) \geq 0$.

To estimate the error we now have

$$x^2y - F_{11}(x,y;x^l,x^u,y^l,y^u) = (x-x^l)(x+x^l)(y-y^l) \quad (146)$$

We want to solve

$$\begin{aligned} \max_{x,y} \quad & (x - x^l)(x + x^l)(y - y^l) \\ \text{subjecto to:} \quad & y \leq y^u + \frac{y^l - y^u}{x^u - x^l}(x - x^l) \\ & x^l \leq x \leq x^u, y^l \leq y \leq y^u \end{aligned} \tag{147}$$

We first claim that the solution of this problem does not occur at an interior point of the domain. To see this note that if we are at an interior point $(\bar{x}, \bar{y})$ of the domain we can increase both $\bar{x}$ and $\bar{y}$ and since the objective function is increasing with both x and y we would not be at an optimal solution. Also it is clear that the optimal solution cannot occur at the faces of the domain $x = x^l$ or $y = y^l$ since in this case the objective function value is 0. We study the objective function at the line $\{(x, y) : y = y^u + \frac{y^l - y^u}{x^u - x^l}(x - x^l)\}$ In this case the problem reduces to

$$\begin{aligned} \max_x \quad & \frac{(x+x^l)(x-x^l)(x^u-x)(y^u-y^l)}{x^u-x^l} \\ \text{subject to} \quad & x^l \leq x \leq x^u \end{aligned} \tag{148}$$

To find the optimal solution we differentiate and find the zero of the derivative which gives

$$x = \frac{1}{3}(x^u + \sqrt{3(x^l)^2 + (x^u)^2}) \tag{149}$$

Plugging this point in the objective function of problem defined in Equation 147 we have that the maximum value is

$$\frac{2(3(x^l)^2(-3x^u + \sqrt{3(x^l)^2 + (x^u)^2}) + (x^u)^2(x^u + \sqrt{3(x^l)^2 + (x^u)^2}))(y^u - y^l)}{27(x^u - x^l)} \tag{150}$$

We now analyze the region where F_{12} is the dominant piece of the underestimator. We already know the region where F_{11} is dominant. Computing $F_{12} - F_{22}$ we have

$$\begin{aligned} & F_{12}(x, y; x^l, x^u, y^l, y^u) - F_{22}(x, y; x^l, x^u, y^l, y^u) \\ & \quad = -(y^u - y^l)x^2 + 2x^l(y^u - y^l)x - (x^u - x^l)^2 y + ((x^l)^2 y^l - 2x^l y^u x^u + (x^u)^2 y^u) \end{aligned} \tag{151}$$

Note that the coefficient of y is negative so if at a point (x, y) the difference in Equation 151 is positive we can increase y and still remain at a point where the difference is positive. If we now solve the equation

$$F_{12}(x, y; x^l, x^u, y^l, y^u) = F_{22}(x, y; x^l, x^u, y^l, y^u) \tag{152}$$

we have that the feasible solutions are the set of points of the form

$$\left\{(x,y) : y = \frac{-(y^u - y^l)x^2 + (x^u)^2 y^u - (x^l)^2 y^l}{(x^u)^2 - (x^l)^2}\right\} \tag{153}$$

Computing the difference from Equation 151 at the line segment joining (x^l, y^u) and (x^u, y^l), the value is given by

$$(x - x^l)(x^u - x)(y^u - y^l) \tag{154}$$

We note that this difference is nonnegative for all the points of the mentioned line. In fact the difference is strictly positive for all the points in the line segment other than the end points. This means that for all the feasible points (x, y) such that

$$\left\{(x,y) : \begin{array}{l} y^u + \frac{y^l - y^u}{x^u - x^l}(x - x^l)\} \leq y \leq \frac{-(y^u - y^l)x^2 + (x^u)^2 y^u - (x^l)^2 y^l}{(x^u)^2 - (x^l)^2} \\ x^l \leq x \leq x^u \end{array}\right\} \tag{155}$$

we have $F_{12}(x, y; x^l, x^u, y^l, y^u) \geq F_{22}(x, y; x^l, x^u, y^l, y^u)$. The expression for the gap for F_{12} is given by

$$\begin{aligned} & x^2 y - F_{12}(x, y; x^l, x^u, y^l, y^u) \\ & \qquad = x^2 y - y^l x^2 - 2x^l(y^u - y^l)x - x^l(2x^u - x^l)y + x^l(2x^u y^u - x^l y^l) \end{aligned} \tag{156}$$

We claim that also in this case the maximum value of the RHS of Equation 156 must be at a boundary point. To see this denote the RHS of Equation 156 by $g(x, y)$. We have

$$\nabla g(x,y) = \begin{bmatrix} 2x(y - y^l) - 2x^l(y^u - y^l) \\ x^2 - x^l(2x^u - x^l) \end{bmatrix} \tag{157}$$

and

$$\nabla^2 g(x,y) = \begin{bmatrix} 2(y - y^l) & 2x \\ 2x & 0 \end{bmatrix} \tag{158}$$

Let λ_1 and λ_2 be the eigenvalues of the matrix in Equation 158. Then by a know result in linear algebra (see for example Theorem 1.2.12 in [51])

$$\begin{aligned} \lambda_1 + \lambda_2 &= \text{trace}(\nabla^2 g(x,y)) = 2(y - y^l) \\ \lambda_1 \cdot \lambda_2 &= \det(\nabla^2 g(x,y)) = -4x^2 \end{aligned} \tag{159}$$

In any point of the interior of the domain we have $2(y-y^l) > 0$ and $-4x^2 < 0$ so we conclude that one of the eigenvalues must be positive and the other must be negative. Using now the second order optimality conditions (see [66], Theorem 2.3 or Proposition 1.1.1 in [26]) we see that the second order necessary optimality conditions do not hold because the Hessian of the objective function is not negative semidefinite so we cannot have a maximum point at the interior of the domain, we conclude that the maximum is attained in one of the curves delimiting the part of the domain where F_{12} is dominant. We have shown before that $F_{12} = F_{11}$ in the line segment joining (x^l, y^u) and (x^u, y^l) and that $F_{12} = F_{22}$ in the curve defined by Equation 153. We computed the maximum error at the line segment and we will show how to compute the maximum error over the curve for the F_{22} function next. Finally for the F_{22} piece the gap is given by

$$\begin{aligned} x^2y - F_{22}(x, y; x^l, x^u, y^l, y^u) &= (x - x^u)(x + x^u)(y - y^u) \\ &= (x^2 - (x^u)^2)(y - y^u) \\ &= ((x^u)^2 - x^2)(y^u - y) \end{aligned} \tag{160}$$

We note that this expression is decreasing with both x and y so the maximum gap will not occur at an interior point of the domain, and by the same reasoning as above it will be found at the curve defined by Equation 152. Plugging the points of the curve in the error expression we get is

$$\frac{(x - x^l)(x + x^l)(x^u - x)(x^u + x)(y^u - y^l)}{(x^u)^2 - (x^l)^2} = \frac{(x^2 - (x^l)^2)((x^u)^2 - x^2)(y^u - y^l)}{(x^u)^2 - (x^l)^2} \tag{161}$$

to find the maximum value in the previous expression we note the maximum is obtained when $x^2 = \frac{(x^l)^2+(x^u)^2}{2}$ (which can be seen by analyzing the behavior of the function $f(z) = (z - a)(b - z)$ which attains the maximum value at $\bar{z} = \frac{a+b}{2}$). Plugging $x^2 = \frac{(x^l)^2+(x^u)^2}{2}$ in the expression in Equation 161 we get that the maximum gap in the region where F_{22} is dominant is given by

$$\frac{(x^u - x^l)(x^u - x^l)(y^u - y^l)}{2} \tag{162}$$

□

Result 8. *The convex hull of x^2y in the box $[x^l, x^u] \times [y^l, y^u]$ is given by*

$$\phi_{conv}(x,y) = \begin{cases} y^l x^2 & y = y^l \\ y^u x^2 & y = y^u \\ f(\hat{x}) & \hat{x} \leq \min\{\tilde{x}, x^u\} \\ f(\tilde{x}) & \tilde{x} = \min\{\tilde{x}, x^u\} < \hat{x} \\ f(u^l) & x^u = \min\{\tilde{x}, x^u\} < \hat{x} \end{cases} \tag{163}$$

where in this case f is defined by

$$f(z) = \frac{(y-y^u)(y-y^l-y^u)}{y-y^l} z^2 + \frac{2x(y-y^u)y^u}{y-y^l} z + \frac{x^2 y^u (y^u - y^l)}{y-y^l} \tag{164}$$

and

$$\hat{x} = \frac{xy^u}{y^u + y^l - y} \tag{165}$$

$$\tilde{x} = \frac{x(y^u - y^l) - x^l(y-y^l)}{y^u - y} \tag{166}$$

For reference

$$\begin{aligned} f(\hat{x}) &= \frac{y^l y^u x^2}{y^u + y^l - y} \\ f(\tilde{x}) &= \frac{(x^l(y-y^l) + xy^l)^2 - ((x^l)^2(y-y^l) + x^2 y^l)y^u}{y - y^u} \\ f(x^u) &= \frac{(x^u)^2(y-y^u)(y-y^l-y^u) + 2xx^u(y-y^u)y^u + x^2 y^u(y^u - y^l)}{y-y^l} \end{aligned} \tag{167}$$

Proof. Note: As mentioned before this result is not new, it is presented in a more general setting in [80]. This result and the proof provide explicit details for this case.

To obtain the convex hull of x^2y in a point $(x,y) \in [x^l, x^u] \times [y^l, y^u]$ we solve the problem:

$$\begin{aligned} \min_{x_a, x_b} \quad & (1-\lambda) y^l x_a^2 + \lambda y^u x_b^2 \\ \text{subject to:} \quad & x = \quad (1-\lambda) x_a + \lambda x_b \\ & y = \quad (1-\lambda) y^l + \lambda y^u \\ & x^l \leq x_a, x_b \leq x^u \\ & 0 \leq \lambda \leq 1 \end{aligned} \tag{168}$$

If $y = y^l$ or $y = y^u$ the result is evident. Consider now $y^l < y < y^u$. First we eliminate x_b from the relation $\lambda x_b = x - (1-\lambda)x_a \Leftrightarrow x_b = \frac{x-(1-\lambda)x_a}{\lambda}$. By using the equality

$y = (1-\lambda)x^l + \lambda x^u$ we get $\lambda = \frac{y-y^l}{y^u-y^l}$ and consequently $x_b = \frac{x_a(y-y^y)+x(y^u-y^l)}{y-y^l}$. So problem in Equation 168 becomes

$$\begin{array}{rl} \min\limits_{x_a} & \frac{(y-y^u)(y-y^l-y^u)}{y-y^l}x_a^2 + \frac{2x(y-y^u)y^u}{y-y^l}x_a + \frac{x^2y^u(y^u-y^l)}{y-y^l} \\ \text{subject to} & x^l \leq \frac{x_a(y-y^y)+x(y^u-y^l)}{y-y^l} \leq x^u \\ & x^l \leq x_a \leq x^u \end{array} \tag{169}$$

or

$$\begin{array}{rl} \min\limits_{x_a} & \frac{(y-y^u)(y-y^l-y^u)}{y-y^l}x_a^2 + \frac{2x(y-y^u)y^u}{y-y^l}x_a + \frac{x^2y^u(y^u-y^l)}{y-y^l} \\ \text{subject to} & \frac{x(y^u-y^l)-x^u(y-y^l)}{y^u-y} \leq x_a \leq \frac{x(y^u-y^l)-x^l(y-y^l)}{y^u-y} \\ & x^l \leq x_a \leq x^u \end{array} \tag{170}$$

To simplify notation we use $f(x_a)$ for the objective function of problem in Equation 170 where f is as defined by Equation 164. The unconstrained minimum of $f(x_a)$ is given by $\hat{x} = \frac{xy^u}{y^u+y^l-y}$. If $\hat{x}$ is feasible the optimal value of problem in Equation 170 is given by $f(\hat{x})$. If $\hat{x}$ is not feasible then (since the coefficient of x_a is positive) the minimum vale is given by

$$\begin{cases} f(\max\{x^l, \frac{x(y^u-y^l)-x^u(y-y^l)}{y^u-y}\}) & \text{if} \quad \hat{x} < \max\{x^l, \frac{x(y^u-y^l)-x^u(y-y^l)}{y^u-y}\} \\ f(\min\{x^u, \frac{x(y^u-y^l)-x^l(y-y^l)}{y^u-y}\}) & \text{if} \quad \hat{x} > \min\{x^u, \frac{x(y^u-y^l)-x^l(y-y^l)}{y^u-y}\} \end{cases} \tag{171}$$

We show that $\hat{x} \geq x^l$ and $\hat{x} \geq \frac{x(y^u-y^l)-x^u(y-y^l)}{y^u-y}$, so the minimum value in problem in Equation 170 is given either given by $f(\hat{x})$ or $f(\min\left\{x^u, \frac{x(y^u-y^l)-x^l(y-y^l)}{y^u-y}\right\})$. We have $\hat{x} = \frac{xy^u}{y^u+y^l-y} \geq \frac{xy^u}{y^u} = x \geq x^l$, since $y^l - y \leq 0$ and $x^l \leq x$. For proving the second inequality we have $\frac{x(y^u-y^l)-x^u(y-y^l)}{y^u-y} - \hat{x} = \frac{(y-y^l)(xy^l+x^u(y-y^l-y^u)}{(y^u-y)(y^u+y^l-y)} \leq 0$ since $y - y^l \geq 0$, $y^u - y \geq 0$, $y^u + y^l - y \geq 0$ and $xy^l + x^u(y - y^l - y^u) = y^l(x - x^u) + x^u(y - y^u) \leq 0$. □

4.5 Convex envelope and underestimators of $-x^2y$ in $[0,1] \times [0,1]$

We continue to denote $S = [0,1] \times [0,1]$ and define $\varphi = -x^2y$.

Result 9. *The convex envelope φ_{conv} of φ in S is given by $\varphi_{conv}(x,y) = -y$.*

Proof. We note that we can use similar techniques to the ones in Result 8 to derive the convex hull of φ, since φ is convex for fixed x and concave for fixed y in S. So the convex

envelope φ_{conv} of φ at a point $(x, y) \in S$ is obtained by solving the following problem:

$$\begin{aligned} \min_{y_a, y_b} \quad & -(1-\lambda)(x^l)^2 y_a - \lambda (x^u)^2 y_b \\ \text{subject to:} \quad & x = \quad (1-\lambda)x^l + \lambda x^u \\ & y = \quad (1-\lambda)y_a + \lambda y_b \\ & 0 \leq y_a, y_b \leq 1 \\ & 0 \leq \lambda \leq 1 \end{aligned} \tag{172}$$

In the present case this reduces to

$$\begin{aligned} \min_{y_a, y_b} \quad & -\lambda y_b \\ \text{subject to:} \quad & x = \quad \lambda \\ & y = \quad (1-\lambda)y_a + \lambda y_b \ \cdot \\ & 0 \leq y_a, y_b \leq 1 \\ & 0 \leq \lambda \leq 1 \end{aligned} \tag{173}$$

By replacing $\lambda y_b = y - (1-\lambda)y_a$ and $\lambda = x$ we get

$$\min_{0 \leq y_a \leq 1} -y + (1-x)y_a \tag{174}$$

so $\varphi_{conv}(x, y) = -y$ since $(1 - x) \geq 0$. □

Result 10. *The convex envelope of $\varphi(x, y) = -x^2 y$ in $[x^l, x^u] \times [y^l, y^u]$ is given by*

$$\begin{aligned} \varphi_{conv}(x, y) &= \max\{l_1(x, y), l_2(x, y)\} \ \textit{with} \\ l_1(x, y) &= -(x^u)^2 y + y^l (x^l + x^u)(x^u - x) \\ l_2(x, y) &= -(x^l)^2 y - y^u (x^l + x^u)(x - x^l) \end{aligned} \tag{175}$$

Proof. Looking again at the problem

$$\begin{aligned} \min_{y_a, y_b} \quad & -(1-\lambda)(x^l)^2 y_a - \lambda (x^u)^2 y_b \\ \text{subject to:} \quad & x = \quad (1-\lambda)x^l + \lambda x^u \\ & y = \quad (1-\lambda)y_a + \lambda y_b \\ & y^l \leq y_a, y_b \leq y^u \end{aligned} \tag{176}$$

we replace $\lambda y_b = y - (1-\lambda)y_a$ and get the problem

$$\begin{aligned} \min_{y_a} \quad & (1-\lambda)\left((x^u)^2 - (x^l)^2\right) y_a - (x^u)^2 y \\ \text{subject to:} \quad & x = \quad (1-\lambda)x^l + \lambda x^u \\ & y^l \leq y_a \leq y^u \\ & \lambda y^l \leq y - (1-\lambda)y_a \leq \lambda y^u \end{aligned} \tag{177}$$

Noting that $\lambda \leq 1$ and $0 \leq x^l \leq x^u$ we have $(1-\lambda)\left((x^u)^2 - (x^l)^2\right) \geq 0$ so the minimum of problem from Equation 177 is attained when $y_a(1-\lambda)$ is minimized. The constraints

$$\begin{aligned} & y^l \leq y_a \leq y^u \\ & \lambda y^l \leq y - (1-\lambda)y_a \leq \lambda y^u \end{aligned} \tag{178}$$

give

$$(1-\lambda)y_a \geq \max\left\{y - \lambda y^u,\ (1-\lambda)y^l\right\} \tag{179}$$

By replacing $(1-\lambda)y_a$ by the right hand side of inequality in Equation 179 and $\lambda = (x - x^l)/(x^u - x^l)$ in Equation 177 we get the result in Equation 175. □

Result 11. *Let $S = [x^l, x^u] \times [y^l, y^u]$. Then*

$$\max_{(x,y)\in S}(-x^2 y - \varphi_{conv}(x,y)) = \begin{cases} \left((x^u)^2 - (x^l)^2\right)\left(y^u - y^l\right) & , \frac{(x^l+x^u)y^l}{2y^u} < x^l \\ \frac{\left((x^l+x^u)y^l - x^l y^u\right)^2}{4y^u} & , x^l \leq \frac{(x^l+x^u)y^l}{2y^u} \leq x^u \end{cases} \tag{180}$$

Proof. We note $\varphi(x,y) - \varphi_{conv}(x,y) = ((x^u)^2 - x^2)y - y^l(x^u - x)(x^l + x^u)$, which is increasing on y. So

$$\begin{aligned} \max_{(x,y)\in S}(\varphi(x,y) - \varphi_{conv}(x,y)) &= \max_{x\in[x^l,x^u]}(\varphi(x,y^u) - \varphi_{conv}(x,y^u)) \\ &= \max_{x\in[x^l,x^u]}\left(-y^u x^2 + \left(x^l + x^u\right)y^l x + \left((x^u)^2 y^u - x^u(x^l + x^u)y^l\right)\right) \end{aligned} \tag{181}$$

Differentiating the expression in Equation 181 with respect to x we find that the point $\hat{x} = \frac{(x^l+x^u)y^l}{2y^u}$ is where the maximum gap is attained (if the point is feasible). In case $\hat{x} < x^l$

maximum gap is at x^l and if $\hat{x} > x^u$ maximum gap is at x^u giving the general expression:

$$\max_{(x,y)\in S}(-x^2y - \varphi_{conv}(x,y)) = \begin{cases} \varphi(x^l, y^u) - \varphi_{conv}(x^l, y^u) & , \hat{x} < x^l \\ \varphi(\hat{x}, y^u) - \varphi_{conv}(\hat{x}, y^u) & , x^l \leq \hat{x} \leq x^u \\ \varphi(x^u, y^u) - \varphi_{conv}(x^u, y^u) & , \hat{x} > x^u \end{cases} \tag{182}$$

But since $\hat{x} = \frac{(x^l+x^u)y^l}{2y^u} \leq \frac{x^l+x^u}{2} \leq x^u$, we never have $\hat{x} > x^u$, so Equation 182 reduces to Equation 180 after expanding the function values computed at $(\hat{x}, y^u)$ and (x^l, y^u). □

4.6 Conclusion

In this chapter we presented a new convex relaxation for cubic terms of the form x^2y. The relaxations are presented in Equation 124 and final form of this relaxation is given in Theorem 3. This is a convex quadratic relaxation that is not as strong as the convex hull but is stronger than the relaxation that would be obtained by using McCormick inequalities in the reformulation of the problem. It is also shown here (Result 10) that the linear relaxations that would be obtained via the McCormick concave envelopes in the reformulated problem would actually give the concave envelope for the term x^2y. However four inequalities would be used in direct relaxation and here we show that two of those suffice. In the next chapter we compare numerical in a branch and bound scheme how the different relaxations behave.

CHAPTER V

NUMERICAL COMPARISON OF CONVEX UNDERESTIMATORS FOR CUBIC POLYNOMIALS PROBLEMS

In this chapter we test numerically how different underestimators for cubic polynomial problems perform in a pure branch and bound scheme. We compare the different schemes against two generated sets of instances, one arising from solving least squares problems, the other having different number of x^2y terms.

5.1 Introduction

We describe here computational results comparing the numerical performance of different approximations for cubic polynomial optimization problems, when used in a branch and bound scheme. The main difference between the different settings tested is the type of approximation used for terms of the type x^2y, that is terms quadratic in x and linear in y. Two main groups of algorithms can be used to solve non-linear optimization problems. The non-linear relaxation can be used in the branch and bound process as the direct analog of the Linear Programing relaxation (see [29], [47], [57], [77]). In the other group of algorithms we have domain decomposition techniques for global optimization techniques (see Horst and Tuy [52], [79]). Here polyhedral relaxations of the non-linear constraints are used to build relaxations for the problem. If the problem is convex the nonlinear relaxation of the problem can also be used as relaxations in each node. Feasible solutions to the original problem are used to prune nodes in the branch and bound process or to improve the polyhedral relaxations. These solutions can be obtained by solving a non-linear relaxation to the original problem or by a local search procedure. Some of the algorithms in this category include outer approximation ([34], [35]), generalized benders decomposition ([40], LP/NLP branch and bound ([70]), and the extended cutting-plane method ([85]). Solvers using these types of relaxations include Bonmin ([28], Couenne ([22]), and Filmint ([10]).

Our approach is conceptually closer to the second group of algorithms, we build polyhedral relaxations for the non-linear constraints, but we also consider non-linear relaxations. The implementations were written in **C** programming Language using **SCIP** (*Solving Constraint Integer Programs*), ([11], [12],[13]). We test the different approximations using two families of instances of problems. Both families of instances were generated for use in this study. One of the families only has linear terms and non-linear terms of the form x^2y, while the other set of instances is generated using least squares so each instance in this family is a general cubic problem.

5.2 Relaxations for Cubic Polynomial Problems

We consider here the optimization of a cubic polynomial problem. A general form of this problem is

$$\begin{aligned} \text{Minimize} \quad & g_0(x) \\ \text{Subject to :} \quad & g_s(x) \leq 0, \quad s \in S \\ & 0 \leq x^l \leq x \leq x^u < \infty \end{aligned} \tag{183}$$

Here S is a finite set and each of the functions $g_i, i \in \{0 \cup S\}$ is a polynomial function of degree up to 3:

$$g_s(x) = a_0^s + \sum_i b_i^s x_i + \sum_{i,j} c_{ij}^s x_i x_j + \sum_{i,j,k} d_{ijk}^s x_i x_j x_k \tag{184}$$

We denote the problem in Equation 183 by $P(l,u)$, its optimal objective function value by $z_{P(l,u)}$) and its feasible region by $D_{P(l,u)}$. We define a relaxation $R(l,u)$ for $P(l,u)$ to be an optimization problem

$$\begin{aligned} \text{Minimize} \quad & G_0(x,y) \\ \text{Subject to :} \quad & G_s(x,y) \leq 0, \quad s \in S' \\ & 0 \leq x^l \leq x \leq x^u < \infty \end{aligned} \tag{185}$$

such that $D_{P(l,u)} \subseteq D_{R(l,u)}$ and $G_0(x) \leq g_0(x), \forall x \in D_{P(l,u)}$. Note that the relaxation may have a different number of constraints than the original problem, and usually extra variables are used in order to define the relaxation. In this case $D_{R(l,p)}$ is not the feasible region of the relaxation but the projection of this feasible region into the variables of the original problem, that is $D_{R(l,p)} = \{x \mid \exists y \text{ such that } G_s(x;y) \leq 0, s \in S', 0 \leq x^l \leq x \leq x^u < \infty\}$.

In particular we have $z_{R(l,u)} \leq z_{P(l,u)}$. The types of relaxations considered here are defined by relaxing each of the terms in the constraints. So given a constraint $g_s(x) \leq 0$ with g_s as defined in Equation 184 we rewrite it for convenience in the form

$$g_s(x) = a_0^s + a_i^s x_i + \sum_{\xi \in \Xi} a_\xi^s \prod_{i \in \xi} x_i \tag{186}$$

where each set $\xi \in \Xi$ has $|\xi| \leq 3$. A relaxation for this constraint is obtained by defining new variables w_ξ and functions $L_\xi^i(x, y; l, u), U_\xi^i(x, y; l, u)$ such that

$$L_\xi^{i_1}(x, y; l, u) \leq \prod_{i \in \xi} x_i \leq U_\xi^{i_2}(x, y; l, u) \quad i_1 \in I_1, i_2 \in I_2 \tag{187}$$

with I_1 and I_2 finite sets. Each of the non-linear terms $\prod_{i \in \xi} x_i$ is replaced by the corresponding variable w_ξ and the constraints

$$\begin{aligned} L_\xi^i(x, y; l, u) - w_\xi \leq 0, \quad i \in I_1 \\ -U_\xi^i(x, y; l, u) + w_\xi \leq 0, \quad i \in I_2 \end{aligned} \tag{188}$$

are added to the relaxation. The different relaxations in this work differ by the way the functions L and U are defined.

5.2.1 Branch and Bound

When problem $P(l, u)$ is solved by a procedure that does not guarantee global optimality we can still use the obtained solution as an upper bound for the global optimum value (as long as the solution is feasible). This happens when we use a non-linear solver to solve the problem. So we consider a pseudo problem $P^a(l, u)$ such that when we say we solve this problem we are in fact solving $P(l, u)$ in such a way that global optimality is not guaranteed but we usually get a feasible solution in the process. In the branch and bound procedure $\mathcal{H}$ denotes the set of active branch-and-bound nodes. Each node k in this set is defined by (l^k, u^k, LB^k) where l^k, u^k are the lower and upper bounds on the variables and LB^k is a known lower bound for the node (that is $LB^k \leq z_{R(l^k, u^k)}$). The global upper bound is denoted by UB. In Algorithm 5 we describe the branch and bound procedure for a general cubic problem.

Set global upper bound $UB := +\infty$
Set $LB^0 := -\infty$
Set node list $\mathcal{H} := \{(l^0, u^0, LB^0)\}$
Set $N_{nodes} = 0$
while $\mathcal{H} \neq \emptyset$ **do**
  Select and remove a node $(l^k, u^k, LB^k) \in \mathcal{H}$
  Update $N_{nodes} := N_{nodes} + 1$
  Solve $R(l^k, u^k)$
  if $R(l^k, u^k)$ *is feasible* **and** $z_{R(l^k,u^k)} < UB$ **then**
    Let (x^k, y^k) be the solution to $R(l^k, u^k)$
    if x^k *is feasible for* $P(l^k, u^k)$ **and** $z_{P(l^k,u^k)} < UB$ **then**
      Let $UB = z_{P(l^k,u^k)}$ /* Fathom by Feasibility */
    else
      /* Branch on x^k */
      Pick $s^* \in S$ such that $g_{s^*}(x^k) > 0$
      Pick i_0 such that approximation of non-linear term with i_0 is inexact.
      Set $l_i = l_i^k$ and $u_i = u_i^k, i \neq i_0$
      Choose $\gamma \in (l_{i_0}^k, u_{i_0}^k)$ and *set* $u_{i_0} = \gamma, l_{i_0} = \gamma$
      Update $\mathcal{H} := \mathcal{H} \cup \{(l, u^k, z_{R(l^k,u^k)}), (l^k, u, z_{R(l^k,u^k)})\}$
    end
    if $N_{nodes} \mod N_0 = 0$ **then**
      Solve $P^a(l^k, u^k)$ and if problem is feasible set solution (x^k, y^k)
      if $z_{P^a(l^k,u^k)} < UB$ **then**
        Update $UB = z_{P^a(l^k,u^k)}$
      end
    end
  end
  Remove each node $(l^k, u^k, LB^k) \in \mathcal{H}$ such that $UB \leq LB^k$
end

Algorithm 5: Generic Branch and Bound for Cubic Problems

5.2.2 Description of Relaxations Considered

Here we describe the relaxations that we consider for cubic polynomial problems. For each of the non-linear terms in a cubic constraint as described by Equation 186 we list the functions $L^i_\xi(x, y; l, u), i \in I_1$ and $U^i_\xi(x, y; l, u), i \in I_2$ that we use for building the relaxation of the non-linear term. Six different settings are considered, s1, s2, s4, s5, s7_k5, and s7_k10 (described in Section 5.4). The approximations differ mainly in two aspects: the type of approximations that are used as lower bounds for the non-linear terms x^2y and the type of solver used in the solution process. As mentioned before in most relaxations each of the non-linear terms is approximated by the same constraints. We list these approximations for all non-linear terms in the model other than x^2y.

5.2.2.1 Lower Bounds

Bilinear Terms (xy)

For the lower bounds we use the convex envelopes on the variables:

$$\begin{aligned} w_{xy} &\geq y^l x + x^l y - x^l y^l \\ w_{xy} &\geq y^u x + x^u y - x^u y^u \end{aligned} \tag{189}$$

Quadratic Terms (x^2)

In this case we pick N_{x^2} points $\bar{x}_i$ in $[x^l, x^u]$ and add the subgradient inequalities

$$w_{x^2} \geq (\bar{x}_i)^2 + 2\bar{x}_i(x - \bar{x}_i) = 2\bar{x}_i x - (\bar{x}_i)^2, \quad i = 1, \ldots, N_{x^2} \tag{190}$$

In this case we pick the following points:

$$\bar{x}_i = x^l + \frac{(x^u - x^l)}{N_{x^2} - 1}(i - 1), \quad i = 1, \ldots, N_{x^2} \tag{191}$$

Cubic Terms (x^3)

For the cubic terms subgradients similar to the ones added for the quadratic case are added to the model:

$$w_{x^3} \geq (\bar{x}_i)^3 + 3(\bar{x}_i)^2(x - \bar{x}_i) = 2\bar{x}_i x - (\bar{x}_i)^2, \quad i = 1, \ldots, N_{x^3} \tag{192}$$

with $N_{x^3} = N_{x^2}$ and $\bar{x}_i$ as defined in Equation 191.

Trilinear Terms (xyz)

We use recursive arithmetic intervals for building the relaxations for the cubic terms. Starting from

$$\begin{aligned} 0 \leq & (x - x^l)(y^u - y)(z - z^l) \\ 0 \leq & (x - x^l)(y - y^l)(z^u - z) \\ 0 \leq & (x^u - x)(y - y^l)(z - z^l) \\ 0 \leq & (x^u - x)(y^u - y)(z^u - z) \end{aligned} \tag{193}$$

we expand each of these inequalities and replace each of the non-linear terms in the expansion by a corresponding variable obtaining linear inequalities which give lower bounds on each trilinear term.

$$\begin{aligned} w_{xyz} \geq & w_{xy}z^l + w_{xz}y^l + x^l w_{yz} - xy^l z^l - x^l yz^l - x^l y^l z + x^l y^l z^l \\ w_{xyz} \geq & w_{xy}z^l + w_{xz}y^u + x^u w_{yz} - xy^u z^l - x^u yz^l - x^u y^u z + x^u y^u z^l \\ w_{xyz} \geq & w_{xy}z^u + w_{xz}y^l + x^u w_{yz} - xy^l z^u - x^u yz^u - x^u y^l z + x^u y^l z^u \\ w_{xyz} \geq & w_{xy}z^u + w_{xz}y^u + x^l w_{yz} - xy^u z^u - y^l yz^u - x^l y^u z + x^l y^u z^u \\ w_{xy} \geq & y^l x + x^l y - x^l y^l \\ w_{xy} \geq & y^u x + x^u y - x^u y^u \\ w_{xz} \geq & z^l x + x^l z - x^l z^l \\ w_{xz} \geq & z^u x + x^u z - x^u z^u \\ w_{yz} \geq & y^l z + z^l y - z^l y^l \\ w_{yz} \geq & y^u z + z^u y - z^u y^u \end{aligned} \tag{194}$$

Quadratic in x, Linear in y Terms $(x^2 y)$

See section 5.3 for details on the different underestimators that we consider for these types of non-linear terms.

5.2.2.2 Upper Bounds

Bilinear Terms (xy)

The concave envelopes are used in this case:

$$\begin{aligned} w_{xy} &\leq y^l x + x^l y - x^l y^l \\ w_{xy} &\leq y^u x + x^u y - x^u y^u \end{aligned} \tag{195}$$

Quadratic Terms (x^2)

We use a secant inequality in to give an upper bound to the function in this case:

$$w_{x^2} \leq (x^l)^2 + \frac{(x^u)^2 - (x^l)^2}{x^u - x^l}(x - x^l) = (x^l + x^u)x - x^l x^u \tag{196}$$

Cubic Terms (x^3)

Secant inequality is also used in this case:

$$w_{x^3} \leq (x^l)^3 + \frac{((x^u)^3 - (x^u)^3)}{x^u - x^l}(x - x^l) \tag{197}$$

Trilinear Terms (xyz)

Recursive arithmetic intervals are used in this case. The upper functions defining the upper bound are in this case:

$$\begin{aligned} w_{xyz} \leq & w_{xy}z^l + x^l w_{yz} + w_{xz}y^u - x^l y^u z - x^l y z^l - x y^u z^l + x^l y^u z^l \\ w_{xyz} \leq & w_{xy}z^u + x^l w_{yz} + w_{xz}y^l - x^l y^l z - x^l y z^u - x y^l z^u + x^l y^l z^u \\ w_{xyz} \leq & w_{xy}z^l + x^u w_{yz} + w_{xz}y^l - x^u y^l z - x^u y z^l - x y^l z^l + x^u y^l z^l \\ w_{xyz} \leq & w_{xy}z^u + x^u w_{yz} + w_{xz}y^u - x^u y^u z - x^u y z^u - x y^u z^u + x^u y^u z^u \\ w_{xy} \leq & y^u x + x^l y - x^l y^u \\ w_{xy} \leq & y^l x + x^u y - x^u y^l \\ w_{xz} \leq & z^u x + x^l z - x^l z^u \\ w_{xz} \leq & z^l x + x^u z - x^u z^l \\ w_{yz} \leq & y^u z + z^l y - z^l y^u \\ w_{yz} \leq & y^l z + z^u y - z^u y^l \end{aligned} \tag{198}$$

Quadratic in x, Linear in y Terms (x^2y)

The two following inequalities are added for the relaxation of this term:

$$\begin{aligned} w_{x^2y} &\leq (x^u)^2y + (x - x^u)(x^l + x^u)y^l \\ w_{x^2y} &\leq (x^l)^2y + (x - x^l)(x^l + x^u)y^u \end{aligned} \tag{199}$$

These inequalities describe the tightest concave set that can be use as upper bounds for these types of non-linear terms.

5.3 Underestimators for x^2y

Like mentioned before the main difference in the approximations is the solver that is used and which functions are chosen as underestimators for the non-linear term x^2y. The first approximation we consider is the convex quadratic approximation introduced in Chapter 4, section 4.4. This approximation is defined by

$$\begin{aligned} w_{x^2y} &\geq y^l x^2 + (x^l)^2 y - (x^l)^2 y^l \\ w_{x^2y} &\geq y^l x^2 + 2x^l(y^u - y^l)x + x^l(2x^u - x^l)y \\ w_{x^2y} &\geq y^u x^2 + (x^u)^2 y - (x^u)^2 y^u \end{aligned} \tag{200}$$

For reference we denote this approximation as *Convex Quadratic Approximation.*

The second type of underestimators we consider for this non-linear term is given the linear functions resulting from applying McCormick inequalities to the non-linear term x^2y. This is done by writing $x^2y = x(xy)$ and first approximating the term in parentheses followed by the approximation of the resulting expression. The resulting inequalities are:

$$\begin{aligned} w_{x^2y} &\geq x^l(x^l(y - 2y^l) + 2xy^l) \\ w_{x^2y} &\geq x^u(x^l(y - 2y^u) + 2xy^u) \\ w_{x^2y} &\geq x^l(-x^l y^l + x^u(y - y^u) + x(y^l + y^u)) \\ w_{x^2y} &\geq x^u(x^l(y - y^l) - x^u y^u + x(y^l + y^u)) \end{aligned} \tag{201}$$

We denote this approximation as *Linear Approximation from McCormick.* The properties of these approximations are studied in Chapter 4. Now we present approximations that are related with the convex envelope of the non-linear term.

5.3.1 Approximations Related with the Convex Envelope of x^2y

In Chapter 4, section 4.4 we presented an explicit formula for the convex envelope of x^2y, ϕ_{conv}, in a rectangle $[x^l, x^u] \times [y^l, y^u]$. This description was first given, in a slightly different form in [80]. The expression for ϕ_{conv} is

$$\phi_{conv}(x,y) = \begin{cases} y^l x^2 & y = y^l \\ y^u x^2 & y = y^u \\ f(\hat{x}) & \hat{x} \leq \min\{\tilde{x}, x^u\} \\ f(\tilde{x}) & \tilde{x} = \min\{\tilde{x}, x^u\} < \hat{x} \\ f(u^l) & x^u = \min\{\tilde{x}, x^u\} < \hat{x} \end{cases} \tag{202}$$

with f defined by Equation 203

$$f(z) = \frac{(y-y^u)(y-y^l-y^u)}{y-y^l} z^2 + \frac{2x(y-y^u)y^u}{y-y^l} z + \frac{x^2 y^u (y^u - y^l)}{y-y^l} \tag{203}$$

and $\hat{x}, \tilde{x}$ given by

$$\begin{aligned} \hat{x} &= \frac{xy^u}{y^u + y^l - y} \\ \tilde{x} &= \frac{x(y^u - y^l) - x^l(y - y^l)}{y^u - y} \end{aligned} \tag{204}$$

This explicit expression of the convex hull of the function is not suitable to directly use as input for a numerical solver. But using the same procedure as in Sahinidis and Tawarmalani [80] we show how this function can be expressed, for each point (x, y), as the optimal value of an optimization problem with conic or semidefinite constraints. If we have $\phi_{conv}(x,y) = \min_\xi\{g(\xi; x, y) \mid h(\xi; x, y) \leq 0\}$, with g and h convex constraints, instead of adding $w_{x^2y} \geq \phi_{conv}(x,y)$, to the relaxations of the cubic problem we can add the constraints $w_{x^2y} \geq g(\xi, x, y)$ and $h(\xi, x, y) \leq 0$ to the relaxation, obtaining a semidefinite or conic quadratic relaxation to the problem. We show for completeness that this is indeed a valid procedure. Consider an optimization problem, $P^1 = \min_x\{f(x) \mid x \in X\}$ which is equivalent to the reformulation $P^2 = \min_{x,z}\{z \mid z \geq f(x), x \in X\}$. Assume now that for each $x \in X$, f can be expressed as an optimization problem $f(x) = \min_\xi\{g(\xi; x) \mid h(\xi; x) \leq 0\}$. We claim that

$$\min_{x,z}\{z \mid z \geq f(x), x \in X\} = \min_{x,z,\xi}\{z \mid z \geq g(\xi, x), h(\xi, x) \leq 0, x \in X\}. \tag{205}$$

Denote the problem in the LHS of Equation 205 by P and the problem on the RHS by R. Let (x^*, z^*) be an optimal solution to P. Then this solution is feasible for problem R and so we should have $z_P \geq z_R$, where z_P and z_R are the optimal solution values of P and R. Let now (x^*, z^*, ξ^*) be an optimal solution to R. We then have $x^* \in X, h(\xi^*, x^*) \leq 0$ and $z^* \geq g(\xi^*, x^*) \geq \min_\xi\{g(\xi, x^*) \mid h(\xi, x^*) \leq 0\} = f(x^*)$ which shows that (x^*, z^*) is a feasible solution to P and completes the proof of the claim.

Looking again at Result 8 from section 4.4 in Chapter 4 we see that we can reformulate the problem giving the value of the convex hull as the following problem:

$$\begin{array}{lllll} \min\limits_{x_p, g_1, g_2} & g_1 + g_2 & & & \\ \text{subject to} & g_1 \cdot (y^u - y) & \geq & y^l x_p^2 (y^u - y^l) & \\ & g_2 \cdot (y - y^l) & \geq & y^u (x - x_p)^2 (y^u - y^l) & \\ & x^l (y^u - y) & \leq & x_p (y^u - y^l) & \leq \quad x^u (y^u - y) \\ & x^l (y - y^l) & \leq & (x - x_p)(y^u - y^l) & \leq \quad x^u (y - y^l) \end{array} \tag{206}$$

where

$$\begin{aligned} g_1 &= \tfrac{y^l (y^u - y^l) x_p^2}{y - y^l} \\ g_2 &= \tfrac{y^u (y^u - y^l)(x - x_p)^2}{y - y^l} \end{aligned} \tag{207}$$

Instead of trying to obtain an explicit expression in x and y we try to derive an easier expression for this problem. From the definition of g_1 and g_2 we have that the first and second constraints in Equation 206 are redundant. The third and fourth inequalities in Equation 206 give linear bounds on x_p :

$$\begin{cases} x_p & \geq & x^l (y^u - y)/(y^u - y^l) \\ x_p & \leq & x^u (y^u - y)/(y^u - y^l) \\ x_p & \geq & x - x^u (y - y^l)/(y^u - y^l) \\ x_p & \leq & x - x^l (y - y^l)/(y^u - y^l) \end{cases} \tag{208}$$

So Equation 206 takes the form:

$$\begin{array}{rrcl} \min_{x_p} & (y^u - y^l)\left(y^u \frac{(x-x_p)^2}{y-y^l} + \frac{y^l x_p^2}{y^u - y}\right) & & \\ \text{subject to} & x_p & \geq & x^l(y^u - y)/(y^u - y^l) \\ & x_p & \leq & x^u(y^u - y)/(y^u - y^l) \\ & x_p & \geq & x - x^u(y - y^l)/(y^u - y^l) \\ & x_p & \leq & x - x^l(y - y^l)/(y^u - y^l) \end{array} \tag{209}$$

Introducing new variables, z_1 and z_2, this is obviously equivalent to the following problem:

$$\begin{array}{rrcl} \min_{x_p, z_1, z_2} & (y^u - y^l)(z_1 + z_2) & & \\ \text{subject to} & z_1 & \geq & y^u \frac{(x-x_p)^2}{y-y^l} \\ & z_2 & \geq & \frac{y^l x_p^2}{y^u - y} \\ & x_p & \geq & x^l(y^u - y)/(y^u - y^l) \\ & x_p & \leq & x^u(y^u - y)/(y^u - y^l) \\ & x_p & \geq & x - x^u(y - y^l)/(y^u - y^l) \\ & x_p & \leq & x - x^l(y - y^l)/(y^u - y^l) \end{array} \tag{210}$$

It is known that the function $f(x, y) = x^2/y, y > 0$ is convex, (see for example the book [50]). It is also easy to show it directly, since this is the perspective function of the convex function $g(x) = x^2$ and therefore we have, with $\lambda \in [0, 1]$ and $y_1, y_2 > 0$:

$$\begin{aligned} & \big(\lambda y_1 + (1-\lambda)y_2\big) g\Big(\frac{\lambda x_1 + (1-\lambda)x_2}{\lambda y_1 + (1-\lambda)y_2}\Big) \\ & = \big(\lambda y_1 + (1-\lambda)y_2\big) g\Big(\frac{\lambda y_1}{\lambda y_1 + (1-\lambda)y_2}\frac{x_1}{y_1} + \frac{(1-\lambda)y_2}{\lambda y_1 + (1-\lambda)y_2}\frac{x_2}{y_2}\Big) \\ & \leq \lambda y_1 g\Big(\frac{x_1}{y_1}\Big) + (1-\lambda)y_2 g\Big(\frac{x_2}{y_2}\Big) \end{aligned} \tag{211}$$

which shows that f is convex if g is convex. It follows that the constraints $z_1 \geq y^u \frac{(x-x_p)^2}{y-y^l}$ and $z_2 \geq \frac{y^l x_p^2}{y^u - y}$ are convex and can be represented either by a linear matrix inequality (LMI) or by a second order cone constraint (SOC). Consider $z_1 \geq y^u \frac{(x-x_p)^2}{y-y^l}$. Since $y^u \geq 0$ and $y - y^l \geq 0$ this constraint is equivalent to

$$\begin{bmatrix} z_1 & \sqrt{y^u}(x - x_p) \\ \sqrt{y^u}(x - x_p) & (y - y^l) \end{bmatrix} \succeq 0 \tag{212}$$

and also

$$\left\| \begin{array}{c} 2\sqrt{y^u}(x-x_p) \\ (y-y^l)-z_1 \end{array} \right\|_2 \leq (y-y^l)+z_1 \tag{213}$$

To see that the LMI in Equation 212 is indeed equivalent to the mentioned inequality recall that a 2×2 matrix $\left(\begin{smallmatrix} a & b \\ b & c \end{smallmatrix}\right)$ is positive definite if $a \geq 0, c \geq 0$ and $ac - b^2 \geq 0$. In the same way the conic constraint in Equation 213 is equivalent to $z_1 \geq y^u \frac{(x-x_p)^2}{y-y^l}$ since given a, b and c the second order conic constraint $\|(a,b)\|_2 \leq c$ is equivalent to having $c \geq 0$ and $a^2 + b^2 \leq c^2$. We have

$$\begin{aligned} &(2\sqrt{y^u}(x-x_p))^2 + ((y-y^l)-z_1)^2 - ((y-y^l)+z_1)^2 \\ &= 4y^u(x-x_p)^2 + ((y-y^l)^2 - 2(y-y^l)z_1 + z_1^2) - ((y-y^l)^2 + 2(y-y^l)z_1 + z_1^2) \\ &= 4y^u(x-x_p)^2 - 4(y-y^l)z_1 \\ &= 4(y^u(x-x_p)^2 - (y-y^l)z_1) \end{aligned} \tag{214}$$

which shows that the two constraint are equivalent.

We could use specialized solvers semidefinite or conic quadratic solvers to build the approximation to the cubic problem. Instead of doing this we focus on the inequality of Equation 213 and develop a polyhedral approximation to this set using the lifted polyhedral approximation to the second order cone developed by Arkadi Nemirovski and Aharon Ben-Tal [24], in a similar fashion to the work on [83]. We recall the approximation for the second order cone provided in [24]. Let $\mathbf{L}^2 = \left\{(x_1, x_2, x_3) \mid \sqrt{x_1^2 + x_2^2} \leq x_3\right\}$. Let ν be a positive integer and $\xi^j, \eta^j, j = 0, \ldots, \nu$ additional variables.

$$\begin{cases} \xi^0 \geq |x_1| \\ \eta^0 \geq |x_2| \end{cases}$$

$$\begin{cases} \xi^j = \cos\left(\frac{\pi}{2^{j+1}}\right)\xi^{j-1} + \sin\left(\frac{\pi}{2^{j+1}}\right)\eta^{j-1} \\ \eta^j \geq \left|-\sin\left(\frac{\pi}{2^{j+1}}\right)\xi^{j-1} + \cos\left(\frac{\pi}{2^{j+1}}\right)\eta^{j-1}\right| \end{cases}, \quad j = 1, \ldots, \nu \tag{215}$$

$$\begin{cases} \xi^\nu \leq x_3 \\ \eta^\nu \leq \tan\left(\frac{\pi}{2^{\nu+1}}\right)\xi^\nu \end{cases}$$

Rewriting the equations and inequalities in Equation 215 as a system of linear homogeneous

inequalities $\prod^{(\nu)}(x_1, x_2, x_3, u) \geq 0$ with u the collection of the $2(\nu+1)$ variables $\xi^j, \eta^j, j = 0, \ldots, \nu$ we have the following result:

Proposition 1 (Nemirovski, Ben-Tal [24] Proposition 2.1). *$\prod^{(\nu)}$ is a polyhedral $\delta(\nu)$ approximation of $\mathbf{L}^2 = \left\{(x_1, x_2, x_3) \mid \sqrt{x_1^2 + x_2^2} \leq x_3\right\}$ with*

$$\delta(\nu) = \frac{1}{\cos\left(\frac{\pi}{2^{\nu+1}}\right)} - 1 = O\left(\frac{1}{4^\nu}\right). \tag{216}$$

where a polyhedral $\varepsilon-$approximation of $\mathbf{L}^k$ is defined as a linear mapping $\prod^k$ such that

1. If $(y, t) \in \mathbf{L}^k$, then there exists u such that $\prod^k(y, t, u) \geq 0$;
2. If (y, t) is such that $\prod^k(y, t, u) \geq 0$ for some u, then $\|y\|_2 \leq (1+\varepsilon)t$

. In the actual implementation we use a modified version of the approximation defined in Equation 215. This approximation is shown in Equation 217 and it is presented in [42].

$$\begin{cases} \alpha_1 = x_1 \cos(\pi) + x_2 \sin(\pi) = -x_1 \\ \beta_1 \geq x_2 \cos(\pi) - x_1 \sin(\pi) = -x_2 \\ -\beta_1 \leq x_2 \cos(\pi) - x_1 \sin(\pi) = -x_2 \end{cases}$$
$$\begin{cases} \alpha_{i+1} = \alpha_i \cos\left(\frac{\pi}{2^i}\right) + \beta_i \sin\left(\frac{\pi}{2^i}\right) \\ \beta_{i+1} \geq \beta_i \cos\left(\frac{\pi}{2^i}\right) - \alpha_i \sin\left(\frac{\pi}{2^i}\right), \quad \forall 2 \leq i < k \\ -\beta_{i+1} \leq \beta_i \cos\left(\frac{\pi}{2^i}\right) - \alpha_i \sin\left(\frac{\pi}{2^i}\right) \end{cases} \tag{217}$$
$$x_3 = \alpha_k \cos\left(\frac{\pi}{2^k}\right) + \beta_k \sin\left(\frac{\pi}{2^k}\right)$$

The last two inequalities in the first and second block are obviously equivalent to $\beta_1 \geq |x_2|$ and $\beta_{i+1} \geq \left|\beta_i \cos\left(\frac{\pi}{2^i}\right) - \alpha_i \sin\left(\frac{\pi}{2^i}\right)\right|$. This representation can be used when we are approximating only the non-negative orthant of the cone $\mathbf{L}^2$. Its advantage with respect to the original approximation is that it uses less constraints and variables. The parameter k in the formulation in Equation 217 corresponds to $\nu + 1$ in the original formulation from Equation 215. We test the lifted approximations using two different settings: $k = 5$ and $k = 10$. We denote these approximations as *Lifted Approximations*. Finally we recall that the constraint $z \geq \frac{x^2}{y}$ is a convex constraint for $y > 0$. Some of the terms in this constraint

are not polynomial and therefore cannot use the constraint directly in the experiments. Multiplying both sides of the inequality by y we get the equivalent inequality

$$zy \geq x^2 \tag{218}$$

This constraint is polynomial but is not convex. We tested how the branch and bound performed using this approximation as the underestimator for the x^2y terms. We denote this setting *Convex Envelope with Bilinear*.

5.4 Description of SCIP Settings to Run Problems

Having described the different constraints and settings that we use in building the different models we now summarize these settings in a concise way so it is easier to reference them in the text. As we have mentioned before the approximations for the terms other than x^2y are the same in all settings so we omit this information from the description. Table 16 summarizes the main differences in the settings. All the instances have a maximum running

Table 16: Summary of different settings

Setting	Solver Type	Relaxation for x^2y
s1	Non-Linear	*Convex Quadratic*
s2	Non-Linear	*McCormick Linear*
s4	Non-Linear	*Convex envelope with bilinear*
s5	Linear	*McCormick Linear*
s7_k5	Linear	*Lifted Approximation*, $k = 5$
s7_k10	Linear	*Lifted Approximation*, $k = 10$

time of one our.

SCIP (*Solving Constraint Integer Programs*), ([13], [11], [12]), is a software package for solving optimization problems. According to the main web page ([2]) for this software "SCIP is currently one of the fastest non-commercial mixed integer programming (MIP) solvers. It is also a framework for constraint integer programming and branch-cut-and-price. It allows total control of the solution process and the access of detailed information down to the guts of the solver." This framework allows the user to develop tailored methods for handling general or specialized classes of problems. A key notion in **SCIP** is the notion of Constraint Handlers. Given a constraint (or group of constraints) we can instruct **SCIP** how handle

this constraint. We can for example specify how the linear programming relaxation of the constraint should look like, or if we add cuts to the problem based on the specific constraint of set of constraints. At the time these experiments were performed **SCIP** had support for general quadratic constraints. The main author for the constraint handler for quadratic constraints is Stefan Vigerske ([25]). In order have **SCIP** solve the cubic optimization problems a constraint handler for cubic constraints was developed based on the code for the quadratic constraint handler. The problems are stored in a modeling language called **ZIMPL** ([56]) which allows to express optimization problems. The syntax of the language resembles the syntax of the **AMPL** ([39]) modeling language. **SOPLEX** ([88]) is the linear solver used for solving the linear problems in the settings for which it applies. Like **SCIP** and **ZIMPL** it is also affiliated with **ZIB** (Zuse Institute Berlin). The software used to solve the non-linear programs is **IPOPT** ([84]) with an additional dependency on **CppAD** ([3]) a library for automatic differentiation written in **C++**. Table 17 gives the versions of the software discussed.

Table 17: Versions of software used

Software	Version
SCIP	2.1.1.3
SOPLEX	1.6.0
ZIMPL	3.2.0
IPOPT	svn trunk (r2158)
CppAD	20120101.3

5.5 Generating Instances With Variable Number of x^2y Terms

The instances described here have only linear terms and non-linear terms of the form x^2y. Only terms with positive coefficients are present in the model so when relaxing these terms only lower approximations are necessary. The input to generate each instance is the number of variables n, the number of constraints m, the number of terms x^2y with positive coefficients N, and a parameter d that gives the fraction of linear terms with respect to the number of variables. Given d the number of linear terms in the generated problem is given by $\lfloor n \cdot d/3 \rfloor$. Given n the maximum value that N can have in this setting is $2 \cdot \binom{n}{2}$ so the

maximum number of terms in the model (either linear or of the form x^2y) is $2 \cdot \binom{n}{2} + n = n^2$. The parameters for the problem generation are summarized in Table 18.

Table 18: Description of problem input

n	number of variables
m	number of constraints
N	total number of x^2y terms with positive coefficients
d	fraction of linear terms in model

The number of terms per constraint is selected randomly in $\{1, \ldots, 2N/m\}$ and if the sum of the number of terms in the constraints is less than the number of terms we add one term to each constraint until the sum of the number of terms in the constraints is equal to the total number of terms. The choice of which x^2y terms are in the model is also random. The choice of the terms that appear in each constraint is made by sampling without replacement from the list of non-linear terms in the model. If the number of terms in a constraint is less than the total number of terms we continue the sampling process from the terms that were not chosen yet. When all the terms are chosen we repeat this process by again sampling without replacement from the original list of terms. The variable bounds are generated uniformly in $[0, 5]$, while the coefficients for the terms are also generated uniformly but in $[0, 10]$. Table 19 gives the parameters used for the problem generation.

Table 19: Listing of Input parameters used in problem generation

n	10, 20, 30, 40, 50
m	5, 10, 20
N	$n^2/2$, $n^2/4$, $n^2/8$
d	0, 1, 2, 3

5.5.1 More Details on Problem Generation

The final form of the problem is

$$\begin{aligned} \min \quad & c'x \\ \text{subject to} \quad & g(x) \leq b \\ & l \leq x \leq u \end{aligned} \tag{219}$$

where $g(\cdot)$ is a cubic polynomial function. When building the problem we generate randomly the coefficients for the function g and the lower and upper bounds for the variables, l and u. The first step is to define a phase I problem, so that the instance will be feasible. This problem is given by

$$\begin{aligned} \min \quad & e's \\ \text{subject to} \quad & g(x) \leq s \\ & l \leq x \leq u \end{aligned} \tag{220}$$

where s are auxiliary variables and e is the $m-$dimensional vector of ones. If $(\bar{x}, \bar{s})$ is a solution to problem in Equation 220 then we set $b_i = \bar{s}_i + 0.3\,|\bar{s}_i|\,, i = 1, \ldots, m$. For finding c we solve a fixed number of the problem instances with uniformly generated values of c using **GAMS** solvers. We then take the coefficients from the instance that required the largest number of nodes to obtain the solution, and solve the corresponding problem in **SCIP** with polyhedral relaxations. If the number of nodes is more than 10, we accept this problem; otherwise, we generate another problem using the same methodology.

5.6 Results From Instances With Variable x^2y Terms

We present here results obtained for the class of problems presented in section 5.5. In Table 20 we report the number of instances for which the primal solution is within the given tolerance of the best solution found across all the instances. The average gaps, first

Table 20: Number of Times Within Tolerance

	Tolerance	s1	s2	s4	s5	s7_k5	s7_k10
Primal	10^{-1}	223	240	210	232	239	239
	10^{-2}	223	240	210	232	239	239
	10^{-3}	222	239	209	232	239	239
	10^{-4}	222	239	209	232	239	239
Dual	10^{-1}	223	210	207	230	225	216
	10^{-2}	221	183	182	212	225	203
	10^{-3}	86	38	71	54	224	194
	10^{-4}	17	3	3	13	223	180

taken over all the problems and second only over the problems that were not successfully solved at the end of time limit are reported in Table 21. A similar table for the time is presented in Table 22 but this time we compute the average time for the problems that were

Table 21: Average Gaps for All Problems and Problems at Time limit

	s1	s2	s4	s5	s7_k5	s7_k10
All	0.16	6.83	1.16	0.52	5.33	191.59
Time Limit	1.46	24.65	4.13	2.96	91.02	1204.98

successfully solved in the end of time limit. Table 23 gives the average number of nodes,

Table 22: Average Time (seconds) for All and Completed Problems

	s1	s2	s4	s5	s7_k5	s7_k10
All	396	1327	1351	677	456	976
Completed	197	465	507	157	260	480

comparing only methods s1, s2, and s4. The averages in each row are computed only for the problems that were successfully solved by both problems in that corresponding row. Finally Table 24 allows us to have an idea of the number of problems that can be solved

Table 23: Average Number of Nodes for Problems Completed Successfully

	s1	s2	s4
s1, s2	159	1255	
s1, s4	275		166

within some specific time limit by each method.

Table 24: Number of Problems Completed by Time

Time (Seconds)	s1	s2	s4	s5	s7_k5	s7_k10
60	145	73	68	157	133	97
120	168	90	89	173	156	110
300	192	114	112	187	178	128
600	211	134	133	192	198	154
1800	220	158	167	198	219	181
3600	227	174	183	205	226	202

5.7 Problem Generation Using Least Squares

We test the proposed methods for upper bounding cubic polynomial problems with a set of problems generated using least squares. The main parameters for the problem generation are in this case the number of variables and the number of constraints, summarized in

Table 25:

Table 25: Number of variables and constraints for the problem generation

Number of variables	5, 10, 20, 50
Number of constraints	5, 10, 15, 20

First we pick m and n the number of constraints and variables in the problem. The problem we want to solve is

$$\begin{array}{rl} \min & \sum_{i=1}^{n} c_i x_i \\ \text{subject to} & g_i(x) \leq b_i, \quad i = 1, \ldots, m \\ & 0 \leq l \leq x \leq u < +\infty \end{array} \tag{221}$$

where each $g_i(\cdot)$ is a cubic polynomial function:

$$g_s(x) = a_0^s + \sum_{i=1}^{n} a_i^{s,l} x_i + \sum_{i,j=1}^{n} a_{ij}^{s,q} x_i x_j + \sum_{i,j,k=1}^{n} a_{ijk}^{s,c} x_i x_j x_k \tag{222}$$

we use the notation $a_i^{s,l}, a_{i,j}^{s,q}, a_{i,j,k}^{s,c}$ to denote the linear, quadratic and cubic coefficients of constraint s, $s = 1, \ldots, m$.

Many of the parameters in the problem are randomly generated. We denote by l_i and u_i the realization of the independent and identically distributed (iid) random variables L_i and U_i where

$$\begin{array}{rcl} L_i & \sim & U(0,1) \\ U_i & \sim & U(l_i, 1) \end{array}, i = 1, \ldots, m \tag{223}$$

Here l_i and u_i are the lower and upper bounds associated with each variable, $l_i \leq x_i \leq u_i$. As we mentioned before we find some of the parameters in this problem by solving a specific least squares problem. Having l and u determined we pick N_p to be a realization of a random variable X where

$$X \sim \lfloor U(50, 126) \rfloor \tag{224}$$

N_p is the number of points that we sample to solve the least squares problem. So we obtain $x^1, \ldots, x^{N_p}$ and $y^1, \ldots, y^{N_p}$, realizations of *iid* random variables $X^1, \ldots, X^{N_p}$ and $Y^1, \ldots, Y^{N_p}$ with

$$\begin{array}{rcl} X_i^s & \sim & U(l_i, u_i) \\ Y^s & \sim & \mathcal{N}(0, 5) \end{array}, i = 1, \ldots, n, \ s = 1, \ldots, N_p \tag{225}$$

The least squares problem is to find $a_0^r, a^{r,l}, a^{r,q}$ and $a^{r,c}$ such that

$$\left\| a_0^r + \sum_{i=1}^{n} a_i^{r,l} x_i + \sum_{i,j=1}^{n} a_{ij}^{r,q} x_i x_j + \sum_{i,j,k=1}^{n} a_{ijk}^{r,c} x_i x_j x_k - y^s \right\|_2^2 \tag{226}$$

is minimized. We note that in Equation 226 the expression is linear in the coefficients so we can find them using standard least squares methods. In matrix form we want to find the least squares solution of the system

$$\begin{bmatrix} 1 & x_1^1 & x_2^1 & \cdots & x_1^1 x_1^1 & \cdots & x_n^1 x_n^1 x_n^1 \\ 1 & x_1^2 & x_2^2 & \cdots & x_1^2 x_1^2 & \cdots & x_n^2 x_n^2 x_n^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 1 & x_1^{N_p} & x_2^{N_p} & \cdots & x_1^{N_p} x_1^{N_p} & \cdots & x_n^{N_p} x_n^{N_p} x_n^{N_p} \end{bmatrix} \begin{bmatrix} a_0^r \\ a_1^{r,l} \\ a_2^{r,l} \\ \vdots \\ a_{11}^{r,q} \\ \vdots \\ a_{nnn}^{r,c} \end{bmatrix} = \begin{bmatrix} y^1 \\ y^2 \\ \vdots \\ y^{N_p} \end{bmatrix} \tag{227}$$

Writing the linear system from Equation 227 in the form $\hat{A}z = \hat{b}$ we are interested in finding a solution $\bar{z}$ that satisfies

$$\left\| \hat{A}\bar{z} - \hat{b} \right\|_2^2 = \min_z \left\| \hat{A}z - \hat{b} \right\|_2^2 \tag{228}$$

The least squares solution is obtained by using **MATLAB** built in function **lsqlin**. We repeat this procedure for each of the constraints and find the coefficients defining the polynomial functions $g_i(\cdot), i = 1, \ldots, m$. The next step is to define an auxiliary problem to find the right hand side of each constraint. The auxiliary problem is

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} s_i \\ \text{subject to} \quad & g_i(x) \leq s_i, \quad i = 1, \ldots, m \\ & 0 \leq l \leq x \leq u < +\infty \end{aligned} \tag{229}$$

Given a solution (x, s) to problem in Equation 229 we define the parameters c_j to be realizations of the iid random variables C_i and b_i to be given by

$$b_i = s_i + |s_i| \, \xi_i^1 \xi_i^2, \; i = 1, \ldots, m \tag{230}$$

with ξ_i^1 and ξ_i^2 realizations of *iid* random variables Ξ_i^1 and Ξ_i^2, respectively. C_i, Ξ_i^1 and Ξ_i^2 are distributed as follows:

$$\begin{aligned} C_j &\sim \mathcal{N}(0,3),\ j = 1,\ldots,n \\ \Xi_1 &\sim -1 + 2\,\text{Bern}(p) \\ \Xi_2 &\sim U(0,0.2) \end{aligned} \tag{231}$$

Bern(p) denotes a Bernoulli random variable with parameter p, that is if $X \sim \text{Bern}(p)$ then $\text{Prob}(\{X = 1\}) = p$ and $\text{Prob}(\{X = 0\}) = 1 - p$. We note that it makes sense not to have the RHS for the problem b be simply the vector s computed in problem in Equation 229 since by the way this problem was designed the resulting feasible set $\{x \mid g_i(x) \leq s_i, i = 1,\ldots,m,\ 0 \leq x \leq u < +\infty\}$ could be quite small, resulting in problems potentially not very interesting. We then solve the resulting problem with **SCIP** using polyhedral underestimators obtained from McCormick envelopes. If the resulting problem takes at least 1000 nodes to solve we are done, otherwise we choose new values for the RHS and for the cost coefficients, again based on Equation 231 and repeat until we are done. If the problem resulting from choosing c and b in Equation 231 is infeasible to many times in a row then we increase p. When we cannot successfully generate a good problem after a finite number of tries then a start the process over again, sampling points and obtaining new coefficients for the constraints.

5.8 Results From Instances Generated From Least Squares Problems

We present results for the class of problems described in section 5.7. The settings for these experiments are the same as for the previous set of problems. These problems are much harder to solve and several methods could not either a primal feasible solution or a dual feasible solution although one exists. In Table 26 we report the number of times a method was not able to find either a primal or dual solution. It can be seen that the when the settings corresponding to Convex Quadratic and Linear from McCormick approximations are used then a dual feasible solution is always found for each of the problems; the same does not apply to the other methods. The results presented in the following tables do not take into account the problems for which a primal or dual solution was not found (so the reported results are only for the "good" instances).

Data: Number of variables, n and number of constraints, m
Result: Cubic Problem
repeat
  Pick l and u, such that $0 \leq l \leq x \leq u < +\infty$
  for $i \in \{1, \ldots, m\}$ **do**
    Pick N_p, number of points in sample to use least squares
    Pick $x^1, \ldots, x^{N_p}$ and $y^1, \ldots, y^{N_p}$
    Build Matrix $\hat{A}$, RHS $\hat{b}$ and solve $\min_z \|\hat{A}z - \hat{b}\|_2^2$ finding $\bar{z}$
    Solve problem in Equation 229 finding s
  end
  $p \longleftarrow \hat{p}$
  $Niter \longleftarrow 0$
  $NNodes \longleftarrow 0$
  while $Niter \leq MaxIter$ *and* $NNodes < 1000$ **do**
    Pick c and compute b according to Equation 230
    Solve problem in Equation 221 obtaining $NNodes$
    Set $NNodes \longleftarrow 0$ if the problem is infeasible
    if *problem is infeasible 3 times in a row* **then**
      $p \longleftarrow \min(1, (1 + \hat{\gamma})p)$
    else if *problem is feasible 3 times in a row* **then**
      $p \longleftarrow (1 - \hat{\gamma})p$
    end
    $Niter \longleftarrow Niter + 1$
  end
until $NNodes >= 1000$

Algorithm 6: Least Squares Problem Generation

Table 26: Number of times method could not find primal or dual solution

	s1	s2	s4	s5	s7_k5	s7_k10
Primal	1	1	7	1	9	8
Dual	0	0	4	0	7	6

Table 27: Number of Times Within Tolerance

	Tolerance	s1	s2	s4	s5	s7_k5	s7_k10
Primal	10^{-1}	68	68	62	77	55	55
	10^{-2}	68	68	62	77	55	55
	10^{-3}	67	68	62	77	55	55
	10^{-4}	60	59	55	65	55	55
Dual	10^{-1}	38	37	32	61	42	31
	10^{-2}	37	35	30	55	42	31
	10^{-3}	15	16	18	35	42	30
	10^{-4}	5	10	10	33	41	30

Table 28: Average Gaps for All Problems and Problems at Time limit

	s1	s2	s4	s5	s7_k5	s7_k10
All	73.60	78.85	86.58	55.53	60.30	80.26
Time Limit	131.17	137.54	144.91	90.95	118.77	149.38

Table 29: Average Time (seconds) for All and Completed Problems

	s1	s2	s4	s5	s7_k5	s7_k10
All	2148	2242	2413	2255	1968	2143
Completed	293	417	650	148	285	451

Table 30: Average Number of Nodes for Problems Completed Successfully

	s1	s2	s4
s1, s2	1747	2822	
s1, s4	1747		1657

5.9 Comments on Numerical Results

Our proposed convex quadratic underestimator has good performance when solving the family of problems with x^2y terms. It is the method that attains that smallest gap, it solves the largest number of instances to optimality, and it is the fastest on average. From Table 24 we can conclude that the Linear McCormick approximation behaves quite well

for the instances that are easier to solve. A total of 187 instances were solved under 5 minutes but only 205 were solved successfully under one hour. This is a difference of only 18 instances, while in the same period the convex quadratic approximation was able to finish solving an extra 35 problems. We compare the number of nodes only for problems solved successfully and only for settings using non-linear solving because in the settings using linear solving **SCIP** would perform some preprocessing to the node which sometimes eliminate the necessity of actually solving the **LP** problem corresponding to the node. This preprocessing is not available when we use non-linear relaxations for the constraints and in this case we can accurately count the nodes in the process (remember that setting s2 is equivalent to the setting s5 except that one of s2 uses the non-linear solver to solve the resulting problems while s5 uses a linear solver). In the case of the lifted approximations we know that in general the number of nodes in the branch and bound tree is larger than the number of nodes if the convex envelope approximation is used. So in this case we compare only the number of nodes using the setting s4 against the setting s1. The results for the number of nodes are according to what is expected, and we can see that in average the number of nodes used by s1 is much closer to the optimal number of nodes than the number of nodes used by s5.

The problems originating from least squares are much harder to solve. In this class of problems the Linear McCormick approximations are the ones that perform better in general. They average gap over all the problems or problems at time limit is the smallest as is the average time taken over all or the completed problems. Here also the average number of nodes coincides to what is expected with the method using the convex envelope taking on average the smallest number of nodes followed by the convex quadratic approximation and finally the Linear from McCormick. An interesting result is regarding the number of completed problems. Here the best performing approach is the lifted linear approximation with $k = 5$. Note that that the average time and average gap of this setting is bigger than the approximations from Linear McCormick.

Table 31: Number of Problems Completed by Time

Time (Seconds)	s1	s2	s4	s5	s7_k5	s7_k10
60	7	3	4	23	33	29
120	15	8	8	24	36	33
300	24	17	20	29	41	37
600	31	27	25	31	43	40
1800	35	34	33	32	48	42
3600	36	35	36	35	49	46

CHAPTER VI

CONCLUSIONS AND FUTURE WORK

In this thesis we have studied optimization of cubic polynomial problems. These are optimization problems for which at least one of the constraints (or the objective function) of the problem is a cubic polynomial and there exists no constraint with order higher than 3. Heuristics for finding good quality feasible solutions and for improving on existing feasible solutions for a complex industrial problem, involving cubic and pooling constraints among other complicating constraints, have been developed. The heuristics for finding feasible solutions are developed based on linear approximations to the original problem that enforce a subset of the original problem constraints while it tries to provide good approximations for the remaining constraints, obtaining in this way nearly feasible solutions. The performance of these heuristics has been tested by using industrial case studies that are of appropriate size, scale and structure. Furthermore, the quality of the solutions can be quantified by comparing the obtained feasible solutions against upper bounds on the value of the problem. Obtaining these upper bounds is an interesting problem by itself, and we have extended efficient existing techniques for bilinear problems for this class of cubic polynomial problems. Despite the efficiency of the upper bound techniques good upper bounds for the industrial case problem could not be computed efficiently within a reasonable time limit (one hour). We have applied the same techniques to subproblems with the same structure but about one fifth of the size and in this case, on average, the gap between the obtained solutions and the computed upper bounds is about 3%.

In the remaining part of the thesis we examined global optimization of cubic polynomial problems with non-negative bounded variables via branch and bound. A theoretical study on the properties of convex underestimators for non-linear terms which are quadratic in one of the variables and linear on the other variable is presented. A new underestimator is introduced for this class of terms. It is also shown that the straightforward polyhedral under

approximation for terms of the form $-x^2y$ is indeed optimal, that is that that approximation defines the convex hull of the term being approximated, although in the straightforward approximation has four constraints and we show that two of these suffice to define the convex hull. This means that there was redundancy in the straightforward approximation, which when eliminated reduces the size of problem possible allowing for a faster solving time.

The final part of the thesis describes the numerical testing of the previously mentioned underestimators together with approximations obtained by considering lifted approximations of the convex hull of the x^2y terms. Two sets of instances are generated for this test and the description of the procedures to generate the instances are detailed here. By analyzing the numerical results we can conclude that our proposed underestimator has the best performance in the family of instances where the only non-linear terms present are of the form x^2y. This makes sense since this underestimator is developed specially for terms of this form. Problems originating from least squares are much harder to solve than the other class of problems. In this class of problems the efficiency of linear programming solvers plays a big role and on average the methods that use these solvers perform better than the others.

As for future work it would be interesting to consider a version of the industrial problem where some of the components are stochastic. This would be interesting from the point of view of both the modeling possibilities that this component with bring (we could try to account for changes of cost in some products, different levels of availability of some of the raw materials, different demand levels of some of the final products), but also for the new solution techniques that would be required in order to solve the resulting problem. In particular decomposition techniques combined with some of the techniques presented here would be an interesting starting point for approaching this problem. For a problem of this scale it would also be interesting to study efficients methods for parallelizing the solution procedure. As for the relaxations of cubic polynomial problems one natural extension of the problem is to consider relaxations where more than one term is relaxed at a time. Recent work addresses this for bilinear problems (see [21]). Another feature of the problem

would be interesting to investigate is the inclusion of the convex quadratic and cubic terms directly in the nonlinear formulations and how this would compare with keeping the current subgradient approximations or building lifted approximations for these convex terms. Still regarding this problem the lifted approximations are not as effective as they promise theoretically. Numerical issues are a concern. It would be interesting to be able to understand if some of these issues can be resolved by special purpose implementations or, on the other hand, provide some kind of explanation, why these approximations do not behave as well as expected for this class of problems.

APPENDIX A

COEFFICIENT MATRICES

A.1 Coefficient Matrices for Property Functions

Due to restrictions we are unable to provide to matrices of coefficients of the polynomials used in the model. When minimizing the average of the property functions the objective has the form

$$\sum_{j,k=1} \omega_{jk} P_{jk}(x_{\cdot j}) \tag{232}$$

where each P_{jk} has the generic form

$$P_{jk}(x_{\cdot j}) = a_0^k + \sum_i b_i^k x_{ji} + \sum_{i_1,i_2} c_{i_1 i_2}^k x_{i_1 j} x_{i_2 j} \sum_{i_1,i_2,i_3} c_{i_1 i_2 i_3}^k x_{i_1 j} x_{i_2 j} x_{i_3 j} \tag{233}$$

We pick one of the final products of the model, $P1$, and report coefficients for weighted averaged model for that final product, and also give the standard deviation from the other final products when compared with $P1$. So in Table 32 the constant coefficient is given by

$$a_0^1 = \frac{\sum_k \omega_{1k} a_0^k}{|N_A|} \tag{234}$$

and the linear coefficient of variable x_i in Table 32 is

$$b_i^1 = \frac{\sum_k \omega_{1k} b_i^k}{|N_A|} \tag{235}$$

Similar formulas are used the obtain the coefficients in Tables 33, 34,and 35. To give an idea of how the coefficients for the other final products differ for these polynomials we give tables with the standard deviation from the mean. To compute the tables with the standard deviation we first compute the average values and use those values to compute the standard deviation. For example for the constant coefficient the average value is computed from $a_0^1, a_0^2, \ldots, a_0^{10}$ and is given by

$$\bar{a}_0 = \frac{\sum_{l=1}^{10} a_0^l}{10} \tag{236}$$

and the standard deviation $\hat{a}_0$ is given by

$$\hat{a}_0 = \sqrt{\frac{\sum_{l=1}^{10}(a_0^l - \bar{a}_0)^2}{9}} \tag{237}$$

The values in Tables 36, 37, 38, and 38 are obtained with similar computations.

Table 32: Constant and Linear Coefficients for P1

Constant Coefficient: 165.9995571719							
Linear Coefficients							
x1	**x2**	**x3**	**x4**	**x5**	**x6**	**x7**	**x8**
3.8421	1.0896	7.2735	2.7595	0.2987	1.8335	-0.9356	1.0174
x9	**x10**	**x11**	**x12**	**x13**	**x14**	**x15**	**x16**
-0.8772	2.9837	-3.5917	-3.3212	10.4237	-0.0478	1.0572	0.0274

A.2 Coefficient Matrices for Structural Properties

The second set of polynomials have the general form

$$P_{jk}(x_{\cdot j}) = \sum_i b_i x_{ji} + \sum_{i_1,i_2} c_{i_1 i_2} x_{i_1 j} x_{i_2 j} \tag{238}$$

Again we cannot show the exact form of the coefficients b_i and $c_{i_1 i_2}$ because of restrictions. We will show the coefficients for a linearized version of these constraints, that is given a point $\tilde{x}$ we build a linear approximation $\tilde{P}_{jk}$ given by

$$\tilde{P}_{jk}(x_{\cdot j}) = P_{jk}(\tilde{x}_{\cdot j}) + \nabla P(\tilde{x}_{\cdot j})'(x_{\cdot j} - \tilde{x}_{\cdot j}) \tag{239}$$

From equation (239) follows that the constant coefficient in the linear approximation is given by

$$P_{jk}(\tilde{x}_{\cdot j}) - \nabla P(\tilde{x}_{\cdot j})'\tilde{x}_{\cdot j} \tag{240}$$

For completeness we note that the i-th component of $\nabla P(\tilde{x}_{\cdot j})$ is given, in terms of the coefficients in equation (238), by

$$\nabla P(\tilde{x}_{\cdot j})_i = b_i + \sum_{l \neq i} c_{il}\tilde{x}_l + 2c_{ii}\tilde{x}_i \tag{241}$$

Tables 40, 41 and, 42 give the constant and linear coefficients for the linear approximation defined in equation 239 when the point considered for the approximation, $\tilde{x}$, is given by

$$\tilde{x} = \frac{\sum_j \tilde{x}_j}{N_{FP}} \tag{242}$$

Table 33: Quadratic Coefficients for P1

	x1	x2	x3	x4	x5	x6	x7	x8
x1	-3.08279							
x2	0.953963	-1.98535						
x3	-3.14273	-0.86258	-1.17427					
x4	-1.02117	0.410447	-0.04515	-0.50009				
x5	0.443476	-0.03685	0.982622		-2.82565			
x6	-0.13225	-0.15066		-0.23745	-0.11261	-2.9062		
x7	-0.04823	-0.23366	0.505696	-0.13682	0.474387	0.206895	-0.198	
x8	0.305472	0.407156	0.237602	0.047117	-0.38092		-0.08128	-1.95472
x9	-0.39764	-0.18849	-0.29447	-0.01153	-0.47092	-0.78589	-0.09835	-0.20212
x10	0.084224	0.262619	-0.1029	-0.17337	-0.08363	0.110019	-0.06049	0.376714
x11	-0.69235	0.310382	-0.0002	0.219528	-0.37365	-0.25878	-0.23359	-0.03076
x12	1.030505	0.515431	-0.30261	0.142211	0.132489	-0.69653	0.207288	0.209419
x13	-0.29317	-0.19394	-0.12472	-0.04026	0.033044	0.590283	-0.32417	0.282681
x14	-1.96156	-0.32058	-0.83081	-0.09854	0.903151	-0.47375	0.18823	0.595637
x15	-2.34452	-0.07601	-1.4873	-1.03884	0.545214	-0.13695	0.432871	0.015972
x16	0.156412	0.420069	-0.11049	0.259538	-0.20544	0.15703	0.695249	-0.00439

	x9	x10	x11	x12	x13	x14	x15	x16
x9	0.110727							
x10	-0.39117	0.469949						
x11	0.532817	0.00524	0.08592					
x12	-0.26861	0.403759	-0.36672	0.639659				
x13	0.448132	0.029476	-0.10657	0.144036	-0.7501			
x14	-0.01337	-0.23086	0.09496	0.948207	0.16734	-0.63498		
x15	-0.33515	-0.4377	0.261844	-0.5626	0.093178	-1.12909	-0.28967	
x16	0.223104	0.323078	-0.03621	-0.26456	-0.0644	-0.50108	-0.49447	0.27529

Table 34: Cubic Coefficients for P1

	x1	x2	x3	x4	x5	x6	x7	x8
x1*x1	0.6266							
x1*x2	-0.1198							
x1*x3	-0.5177							
x1*x12	1.0189							
x1*x14	1.1305							
x1*x15	1.9074							
x2*x2		1.5943						
x2*x3	0.1785							
x2*x4	-0.0827							
x2*x12	0.3659	0.5660						
x2*x13	-0.9367							
x2*x14	-0.1015	0.5189						
x2*x15	-1.0422	0.7328						
x3*x3	0.4995	-0.7037	0.0882					
x3*x4	-0.3429	-0.0234						
x3*x12	-0.1347	0.0423	-0.5868					
x3*x13	-0.2891	0.2691						
x3*x14	0.6229	0.5155	0.8194					
x3*x15	1.1342	0.1387	0.3968					
x4*x4				-1.0956				
x4*x12	-0.0144	0.0398	-0.2121					
x4*x13	-0.3327	0.2729	-0.4715					
x4*x14	0.4557	-0.0771	0.0669					
x4*x15	0.2675	0.6358	0.6123					
x5*x5					3.0474			
x6*x6						1.9755		
x7*x7							0.7976	
x7*x8							0.9195	
x7*x9							0.5681	
x7*x16							-0.2723	
x8*x8							1.6050	-0.5966
x8*x9								1.6263
x8*x16							-0.7737	-0.1171
x9*x9							-0.2435	0.4000
x9*x16							0.6351	-0.1146
x12*x13	0.3605	0.0886	-0.5675	0.0521				
x12*x14	-0.2202	-0.1852	-0.1853	-0.3107				
x12*x15	-0.4281	-0.1413	0.1856	0.8005				
x13*x14	0.0436		0.1591	0.3413				
x13*x15	-0.4464	-0.1120	-0.2429	-0.6299				
x14*x14	-0.0319		0.3149					
x14*x15	-0.2687	0.1977	-0.0677	-0.2976				
x15*x15	1.0396	0.5562	0.8746					
x16*x16							-0.6753	0.4959

Table 35: Cubic Coefficients for P1, continued

	x9	**x10**	**x12**	**x13**	**x14**	**x15**	**x16**
x9*x9	0.2277						
x9*x16	-1.0231						
x10*x10		-0.4308					
x12*x12			0.7439				
x13*x13				-1.4818			
x13*x14			-0.4504				
x13*x15			-0.2472				
x14*x14			-0.9687		0.4017		
x14*x15			-0.2024	-0.6592	0.4709		
x15*x15			-0.1602		0.2126	0.0132	
x16*x16	-0.3019						1.2892

Table 36: Standard Deviation for Constant and Linear Coefficients

Constant Coefficient: 139.3115							
Linear Coefficients							
x1	**x2**	**x3**	**x4**	**x5**	**x6**	**x7**	**x8**
3.2244	0.9144	6.1041	2.3158	0.2507	1.5387	0.7852	0.8539
x9	**x10**	**x11**	**x12**	**x13**	**x14**	**x15**	**x16**
0.7361	2.5040	3.0143	2.7873	8.7479	0.0401	0.8872	0.0230

where in this case $\tilde{x}_j$ is the benchmark solution for final product j.

Table 37: Standard Deviation for Quadratic Coefficients

	x1	x2	x3	x4	x5	x6	x7	x8
x1	2.5872							
x2	0.8006	1.6662						
x3	2.6375	0.7239	0.9855					
x4	0.8570	0.3445	0.0379	0.4197				
x5	0.3722	0.0309	0.8246		2.3714			
x6	0.1110	0.1264		0.1993	0.0945	2.4390		
x7	0.0405	0.1961	0.4244	0.1148	0.3981	0.1736	0.1662	
x8	0.2564	0.3417	0.1994	0.0395	0.3197		0.0682	1.6405
x9	0.3337	0.1582	0.2471	0.0097	0.3952	0.6595	0.0825	0.1696
x10	0.0707	0.2204	0.0864	0.1455	0.0702	0.0923	0.0508	0.3161
x11	0.5810	0.2605	0.0002	0.1842	0.3136	0.2172	0.1960	0.0258
x12	0.8648	0.4326	0.2540	0.1193	0.1112	0.5845	0.1740	0.1758
x13	0.2460	0.1628	0.1047	0.0338	0.0277	0.4954	0.2721	0.2372
x14	1.6462	0.2690	0.6972	0.0827	0.7579	0.3976	0.1580	0.4999
x15	1.9676	0.0638	1.2482	0.8718	0.4576	0.1149	0.3633	0.0134
x16	0.1313	0.3525	0.0927	0.2178	0.1724	0.1318	0.5835	0.0037

	x9	x10	x11	x12	x13	x14	x15	x16
x9	0.0929							
x10	0.3283	0.3944						
x11	0.4472	0.0044	0.0721					
x12	0.2254	0.3388	0.3078	0.5368				
x13	0.3761	0.0247	0.0894	0.1209	0.6295			
x14	0.0112	0.1937	0.0797	0.7958	0.1404	0.5329		
x15	0.2813	0.3673	0.2197	0.4722	0.0782	0.9476	0.2431	
x16	0.1872	0.2711	0.0304	0.2220	0.0540	0.4205	0.4150	0.2310

Table 38: Standard Deviation for Cubic Coefficients

	x1	x2	x3	x4	x5	x6	x7	x8
x1*x1	0.5259							
x1*x2	0.1005							
x1*x3	0.4344							
x1*x12	0.8551							
x1*x14	0.9488							
x1*x15	1.6007							
x2*x2		1.3380						
x2*x3	0.1498							
x2*x4	0.0694							
x2*x12	0.3071	0.4750						
x2*x13	0.7861							
x2*x14	0.0851	0.4355						
x2*x15	0.8747	0.6149						
x3*x3	0.4192	0.5906	0.0740					
x3*x4	0.2877	0.0197						
x3*x12	0.1130	0.0355	0.4924					
x3*x13	0.2426	0.2259						
x3*x14	0.5228	0.4326	0.6877					
x3*x15	0.9518	0.1164	0.3330					
x4*x4				0.9194				
x4*x12	0.0121	0.0334	0.1780					
x4*x13	0.2792	0.2291	0.3957					
x4*x14	0.3824	0.0647	0.0561					
x4*x15	0.2245	0.5336	0.5139					
x5*x5					2.5575			
x6*x6						1.6579		
x7*x7							0.6693	
x7*x8							0.7717	
x7*x9							0.4768	
x7*x16							0.2285	
x8*x8							1.3469	0.5006
x8*x9								1.3649
x8*x16							0.6493	0.0983
x9*x9							0.2044	0.3357
x9*x16							0.5330	0.0962
x12*x13	0.3025	0.0744	0.4763	0.0437				
x12*x14	0.1848	0.1555	0.1555	0.2607				
x12*x15	0.3592	0.1186	0.1558	0.6718				
x13*x14	0.0366		0.1335	0.2865				
x13*x15	0.3747	0.0940	0.2039	0.5286				
x14*x14	0.0267		0.2642					
x14*x15	0.2255	0.1659	0.0568	0.2498				
x15*x15	0.8725	0.4667	0.7340					
x16*x16							0.5667	0.4162

Table 39: Standard Deviation for Cubic Coefficients, continued

	x9	x10	x12	x13	x14	x15	x16
x9*x9	0.1911						
x9*x16	0.8586						
x10*x10		0.3615					
x12*x12			0.6243				
x13*x13				1.2436			
x13*x14			0.3780				
x13*x15			0.2075				
x14*x14			0.8130		0.3371		
x14*x15			0.1699	0.5532	0.3952		
x15*x15			0.1344		0.1784	0.0111	
x16*x16	0.2534						1.0819

Table 40: Constant Coefficients for the Stability Model

Constant Coefficiens					
P11	**P12**	**P13**	**P14**	**P15**	**P16**
54788.9473	-46605.4572	-249.8654	-8580.3181	666.6916	-687.7796
P17	**P18**	**P19**			
1.4449	1.4158	106.0332			

Table 41: Linear Coefficients for the Stability Model

	P11	P12	P13	P14	P15	P16
x1	0.4659	-0.1552	0.2144	0.0113	-0.0016	-0.0023
x2	-0.4576	0.4766	-0.0367	0.0484	-0.0281	0.0153
x3	0.0964	-0.1385	-0.1664	0.1026	-0.0305	0.0525
x4	-0.2111	-0.0291	0.0200	-0.0494	0.2052	-0.1497
x7	2.5382	-2.7376	-0.3886	1.0695	0.0192	-0.0593
x8	0.0000	0.2633	-0.0795	-0.0404	0.1615	-0.1064
x9	-1.3607	1.2166	-0.3003	0.3117	0.0402	-0.0543
x10	5.2373	-4.3147	0.5221	0.7625	-0.0713	0.2807
x17	-0.3613	0.3151	-0.1264	0.0938	0.0340	-0.0449
x18	-6.1047	4.7069	-2.4758	0.7448	0.1777	-0.0218
x19	-6.1573	4.8493	-1.1156	0.2194	-0.2197	0.2138
x20	-1.4805	1.3490	-0.1442	-0.0159	-0.1701	0.1967
x21	-6.2224	1.5092	-2.7868	0.0520	-0.1657	-0.0008
x22	-0.2048	-0.0928	-0.1458	0.0537	-0.1061	0.0690
x23	-1.1133	1.1738	-0.1989	0.2169	0.0861	-0.1706
x24	-0.6032	0.7100	0.0581	0.2585	-0.1757	0.2184
x25	-1.6171	0.7808	-0.6738	-0.3204	0.2867	-0.3391
x26	-2.4487	2.1269	-0.5557	-0.0066	0.2067	-0.2609

Table 42: Linear Coefficients for the Stability Model

	P17	P18	P19
x1	0.0057	0.0014	0.1040
x2	-0.0081	-0.0031	-0.0745
x3	0.0156	-0.0072	0.1040
x4	0.0427	-0.0049	-0.7713
x7	-0.1295	0.1681	0.3576
x8	0.0253	-0.0643	0.0802
x9	-0.0140	0.0463	0.0956
x10	-0.2104	0.2246	0.0602
x17	0.0002	-0.0011	0.0560
x18	0.0056	0.0309	-0.0320
x19	-0.0795	0.0681	-0.0235
x20	-0.0106	0.0013	0.4502
x21	-0.1379	0.0745	0.9212
x22	-0.0326	0.0054	0.0540
x23	-0.0036	-0.0280	0.0465
x24	0.0245	-0.0196	0.8924
x25	-0.0002	-0.0462	0.1051
x26	-0.0158	0.0584	-0.0258

REFERENCES

[1] `http://www.gams.com`.

[2] `http://scip.zib.de/scip.shtml`.

[3] `http://www.coin-or.org/CppAD/Doc/cppad.xml`.

[4] *GAMS/COIN Solver Manual*. GAMS Development Corporation.

[5] *GAMS/CONOPT Solver Manual*. GAMS Development Corporation.

[6] *GAMS/CPLEX 12 Solver Manual*. GAMS Development Corporation.

[7] *GAMS/DICOPT Solver Manual*. GAMS Development Corporation.

[8] *GAMS/SNOPT Solver Manual*. GAMS Development Corporation.

[9] *GAMS — A User's Guide*. Washington, DC, USA: GAMS Development Corporation, 2010.

[10] Abhishek, K., Leyffer, S., and Linderoth, J., "Filmint: An outer approximation-based solver for convex mixed-integer nonlinear programs," *INFORMS Journal on computing*, vol. 22, no. 4, pp. 555–567, 2010.

[11] Achterberg, T., "Scip: solving constraint integer programs," *Mathematical Programming Computation*, vol. 1, no. 1, pp. 1–41, 2009.

[12] Achterberg, T., Berthold, T., Koch, T., and Wolter, K., "Constraint integer programming: A new approach to integrate cp and mip," *Integration of AI and OR techniques in constraint programming for combinatorial optimization problems*, pp. 6–20, 2008.

[13] Achterberg, T., *Constraint Integer Programming*. PhD thesis, 2009.

[14] Adhya, N., Tawarmalani, M., and Sahinidis, N., "A lagrangian approach to the pooling problem," *Industrial & engineering chemistry research*, vol. 38, no. 5, pp. 1956–1972, 1999.

[15] Adjiman, C., Androulakis, I., and Floudas, C., "A global optimization method,[alpha] bb, for general twice-differentiable constrained nlps–ii. implementation and computational results," *Computers & Chemical Engineering*, vol. 22, no. 9, pp. 1159–1179, 1998.

[16] Adjiman, C., Dallwig, S., Floudas, C., and Neumaier, A., "A global optimization method,[alpha] bb, for general twice-differentiable constrained nlps–i. theoretical advances," *Computers & Chemical Engineering*, vol. 22, no. 9, pp. 1137–1158, 1998.

[17] Al-Khayyal, F. and Falk, J., "Jointly constrained biconvex programming," *Mathematics of Operations Research*, vol. 8, no. 2, pp. 273–286, 1983.

[18] Al-Khayyal, F. and Sherali, H., "On finitely terminating branch-and-bound algorithms for some global optimization problems," *SIAM Journal on Optimization*, vol. 10, no. 4, pp. 1049–1057, 2000.

[19] Bagajewicz, M., "A review of recent design procedures for water networks in refineries and process plants," *Computers & Chemical Engineering*, vol. 24, no. 9, pp. 2093–2113, 2000.

[20] Baker, T. and Lasdon, L., "Successive linear programming at exxon," *Management Science*, vol. 31, no. 3, pp. 264–274, 1985.

[21] Bao, X., Sahinidis, N., and Tawarmalani, M., "Multiterm polyhedral relaxations for nonconvex, quadratically constrained quadratic programs," *Optimization Methods & Software*, vol. 24, no. 4-5, pp. 485–504, 2009.

[22] Belotti, P., "Couenne: a users manual," *Lehigh University*, 2009.

[23] Ben-Tal, A., Eiger, G., and Gershovitz, V., "Global minimization by reducing the duality gap," *Math. Program.*, vol. 63, pp. 193–212, 1994.

[24] Ben-Tal, A. and Nemirovski, A., "On polyhedral approximations of the second-order cone," *Math. Oper. Res.*, vol. 26, no. 2, pp. 193–205, 2001.

[25] Berthold, T., Heinz, S., and Vigerske, S., "Extending a cip framework to solve miqcps," in *Mixed Integer Nonlinear Programming* (Lee, J. and Leyffer, S., eds.), vol. 154 of *The IMA Volumes in Mathematics and its Applications*, pp. 427–444, Springer New York, 2012.

[26] Bertsekas, D. P., *Nonlinear Programming*. Belmont, Massachusetts: Athena Scientific, second ed., 1999.

[27] Bixby, R., "Implementing the simplex method: The initial basis," *ORSA Journal on Computing*, vol. 4, no. 3, pp. 267–284, 1992.

[28] Bonami, P., Biegler, L., Conn, A., Cornuéjols, G., Grossmann, I., Laird, C., Lee, J., Lodi, A., Margot, F., Sawaya, N., and others, "An algorithmic framework for convex mixed integer nonlinear programs," *Discrete Optimization*, vol. 5, no. 2, pp. 186–204, 2008.

[29] Borchers, B. and Mitchell, J., "An improved branch and bound algorithm for mixed integer nonlinear programs," *Computers & Operations Research*, vol. 21, no. 4, pp. 359–367, 1994.

[30] Cafieri, S., Lee, J., and Liberti, L., "On convex relaxations of quadrilinear terms," *Journal of Global Optimization*, vol. 47, no. 4, pp. 661–685, 2010.

[31] Dantzig, G., *Linear Programming and Extensions*. Princeton University Press, 1963.

[32] DeWitt, C., Lasdon, L., Waren, A., Brenner, D., and Melhem, S., "Omega: An improved gasoline blending system for texaco," *Interfaces*, vol. 19, no. 1, pp. 85–101, 1989.

[33] Drud, A., "Conopt - a large-scale grg code," *ORSA Journal on Computing*, vol. 6, no. 2, pp. 207–216, 1994.

[34] Duran, M. and Grossmann, I., "An outer-approximation algorithm for a class of mixed-integer nonlinear programs," *Mathematical programming*, vol. 36, no. 3, pp. 307–339, 1986.

[35] Fletcher, R. and Leyffer, S., "Solving mixed integer nonlinear programs by outer approximation," *Mathematical Programming*, vol. 66, no. 1, pp. 327–349, 1994.

[36] Floudas, C., *Deterministic global optimization: theory, methods and applications*, vol. 37. Springer, 1999.

[37] Floudas, C. and Aggarwal, A., "A decomposition strategy for global optimum search in the pooling problem," *ORSA Journal on Computing*, vol. 2, no. 3, pp. 225–235, 1990.

[38] Floudas, C., Aggarwal, A., and Ciric, A., "Global optimum search for nonconvex nlp and minlp problems," *Computers & chemical engineering*, vol. 13, no. 10, pp. 1117–1132, 1989.

[39] Fourer, R., Gay, D., and Kernighan, B., *Ampl*. Scientific Press, 1993.

[40] Geoffrion, A., "Generalized benders decomposition," *Journal of optimization theory and applications*, vol. 10, no. 4, pp. 237–260, 1972.

[41] Gill, P., Murray, W., and Saunders, M., "Snopt: An sqp algorithm for large-scale constrained optimization," *SIAM journal on optimization*, vol. 12, no. 4, pp. 979–1006, 2002.

[42] Glineur, F. and others, "Computational experiments with a linear approximation of second-order cone optimization," 2000.

[43] Gounaris, C. and Floudas, C., "Tight convex underestimators for-continuous problems: I. univariate functions," *Journal of Global Optimization*, vol. 42, no. 1, pp. 51–67, 2008.

[44] Gounaris, C. and Floudas, C., "Tight convex underestimators for-continuous problems: Ii. multivariate functions," *Journal of Global Optimization*, vol. 42, no. 1, pp. 69–89, 2008.

[45] Gounaris, C. E., Misener, R., and Floudas, C. A., "Computational comparison of piecewise linear relaxations for pooling problems," *Industrial & Engineering Chemistry Research*, vol. 48, no. 12, pp. 5742–5766, 2009.

[46] Greenberg, H. J., "Analyzing the Pooling Problem," *INFORMS JOURNAL ON COMPUTING*, vol. 7, no. 2, pp. 205–217, 1995.

[47] Gupta, O. and Ravindran, A., "Branch and bound experiments in convex nonlinear integer programming," *Management Science*, vol. 31, no. 12, pp. 1533–1546, 1985.

[48] Haverly, C. A., "Studies of the behavior of recursion for the pooling problem," *SIGMAP Bull.*, pp. 19–28, December 1978.

[49] Haverly, C. A., "Behavior of recursion model - more studies," *SIGMAP Bull.*, pp. 22–28, April 1979.

[50] Hiriart-Urruty, J. and Lemaréchal, C., *Fundamentals of convex analysis*. Springer Verlag, 2001.

[51] Horn, R. A. and Johnson, C. R., *Matrix Analysis*. Cambridge University Press, 1990.

[52] Horst, R. and Tuy, H., *Global Optimization : Deterministic Approaches (Second, Revised Edition)*. Berlin: Springer-Verlag, 1992.

[53] Jach, M., Michaels, D., and Weismantel, R., "The convex envelope of (n-1)-convex functions," *SIAM Journal on Optimization*, vol. 19, no. 3, pp. 1451–1466, 2008.

[54] Keha, A. B., de Farias Jr., I. R., and Nemhauser, G. L., "Models for representing piecewise linear cost functions," *Operations Research Letters*, vol. 32, no. 1, pp. 44 – 48, 2004.

[55] Kilinc-Karzan, F., *Tractable relaxations and efficient algorithmic techniques for large-scale optimization*. PhD thesis, 2011.

[56] Koch, T., *Rapid Mathematical Programming*. PhD thesis, Technische Universität Berlin, 2004. ZIB-Report 04-58.

[57] Leyffer, S., "Integrating sqp and branch-and-bound for mixed integer nonlinear programming," *Computational Optimization and Applications*, vol. 18, no. 3, pp. 295–309, 2001.

[58] Li, H., Tsai, J., and Floudas, C., "Convex underestimation for posynomial functions of positive variables," *Optimization letters*, vol. 2, no. 3, pp. 333–340, 2008.

[59] Lu, H., Li, H., Gounaris, C., and Floudas, C., "Convex relaxation for solving posynomial programs," *Journal of Global Optimization*, vol. 46, no. 1, pp. 147–154, 2010.

[60] Maranas, C. and Floudas, C., "Finding all solutions of nonlinearly constrained systems of equations," *Journal of Global Optimization*, vol. 7, no. 2, pp. 143–182, 1995.

[61] McCarl, B. A., Meeraus, A., Eijk, P. V. D., Bussieck, M., Dirkse, S., Steacy, P., and Nelissen, F., *McCarl GAMS User Guide*. GAMS Development Corporation, 2011.

[62] McCormick, G. P., "Computability of global solutions to factorable nonconvex programs: Part i - convex underestimating problems," *Mathematical Programming*, vol. 10, pp. 147–175, 1976. 10.1007/BF01580665.

[63] Meyer, C. and Floudas, C., "Global optimization of a combinatorially complex generalized pooling problem," *AIChE journal*, vol. 52, no. 3, pp. 1027–1037, 2005.

[64] Misener, R. and Floudas, C. A., "Advances for the pooling problem: Modeling, global optimization, and computational studies," *Applied and Computational Mathematics*, vol. 8, no. 1, pp. 3–22, 2009.

[65] Misener, R., Gounaris, C. E., and Floudas, C. A., "Global optimization of gas lifting operations: A comparative study of piecewise linear formulations," *Industrial & Engineering Chemistry Research*, vol. 48, no. 13, pp. 6098–6104, 2009.

[66] Nocedal, J. and Wright, S. J., *Numerical Optimization*. New York: Springer, 1999.

[67] Padberg, M., "Approximating separable nonlinear functions via mixed zero-one programs," *Oper. Res. Lett.*, vol. 27, pp. 1–5, Aug. 2000.

[68] Pardalos, P. and Vavasis, S., "Quadratic programming with one negative eigenvalue is np-hard," *Journal of Global Optimization*, vol. 1, no. 1, pp. 15–22, 1991.

[69] Pham, V., Laird, C., and El-Halwagi, M., "Convex hull discretization approach to the global optimization of pooling problems," *Industrial & Engineering Chemistry Research*, vol. 48, no. 4, pp. 1973–1979, 2009.

[70] Quesada, I. and Grossmann, I., "An lp/nlp based branch and bound algorithm for convex minlp optimization problems," *Computers & chemical engineering*, vol. 16, no. 10, pp. 937–947, 1992.

[71] Rigby, B., Lasdon, L., and Waren, A., "The evolution of texacos blending systems: From omega to starblend," *Interfaces*, vol. 25, no. 5, pp. 64–83, 1995.

[72] Rockafellar, R., *Convex analysis*, vol. 28. Princeton university press, 1996.

[73] Ryoo, H. S. and Sahinidis, N. V., "Analysis of bounds for multilinear functions," *Journal of Global Optimization*, vol. 19, pp. 403–424, 2001. 10.1023/A:1011295715398.

[74] Sahinidis, N. V., *BARON: Global Optimization of Mixed-Integer Nonlinear Programs,* User's Manual, 2012.

[75] Sahni, S., "Computationally related problems," *SIAM Journal on Computing*, vol. 3, no. 4, pp. 262–279, 1974.

[76] Sherali, H. and Tuncbilek, C., "Comparison of two reformulation-linearization technique based linear programming relaxations for polynomial programming problems," *Journal of Global Optimization*, vol. 10, no. 4, pp. 381–390, 1997.

[77] Stubbs, R. and Mehrotra, S., "A branch-and-cut method for 0-1 mixed convex programming," *Mathematical Programming*, vol. 86, no. 3, pp. 515–532, 1999.

[78] Tawarmalani, M. and Sahinidis, N. V., "A polyhedral branch-and-cut approach to global optimization," *Mathematical Programming*, vol. 103, pp. 225–249, 2005.

[79] Tawarmalani, M. and Sahinidis, N., "Global optimization of mixed-integer nonlinear programs: A theoretical and computational study," *Mathematical programming*, vol. 99, no. 3, pp. 563–591, 2004.

[80] Tawarmalani, M. and Sahinidis, N. V., "Semidefinite relaxations of fractional programs via novel convexification techniques," *J. of Global Optimization*, vol. 20, no. 2, pp. 133–154, 2001.

[81] Tawarmalani, M. and Sahinidis, N. V., "Convex extensions and envelopes of lower semi-continuous functions," *Math. Program., Ser. A*, vol. 93, pp. 247–263, 2002.

[82] Tawarmalani, M. and Sahinidis, N. V., *Convexification and Global Optimization in Continuous And*. Norwell, MA, USA: Kluwer Academic Publishers, 2002.

[83] Vielma, J. P., Ahmed, S., and Nemhauser, G. L., "A lifted linear programming branch-and-bound algorithm for mixed-integer conic quadratic programs," *INFORMS Journal on Computing*, vol. 20, no. 3, pp. 438–450, 2008.

[84] Wchter, A. and Biegler, L. T., "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, pp. 25–57, 2006.

[85] Westerlund, T. and Pettersson, F., "An extended cutting plane method for solving convex minlp problems," *Computers & Chemical Engineering*, vol. 19, pp. 131–136, 1995.

[86] Wicaksono, D. S. and Karimi, I. A., "Piecewise milp under- and overestimators for global optimization of bilinear programs," *AIChE Journal*, vol. 54, no. 4, pp. 991–1008, 2008.

[87] Wolfram, S., *The MATHEMATICA® Book, Version 4*. Cambridge university press, 1999.

[88] Wunderling, R., "Paralleler und objektorientierter simplex," Tech. Rep. TR-96-09, ZIB, Takustr.7, 14195 Berlin, 1996.